

```

EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD FFFFFFFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD FFFFFFFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD FFFFFFFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEEEEEEEEEEEEEEEEE DDD DDD FFF
EEEEEEEEEEEEEEEEEE DDD DDD FFFFFFFF
EEEEEEEEEEEEEEEEEE DDD DDD FFFFFFFF
EEEEEEEEEEEEEEEEEE DDD DDD FFFFFFFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD FFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD FFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD FFF

```

```

EEEEEEEEEE DDDDDDDD FFFFFFFF DDDDDDDD EEEEEEEEE SSSSSSSS IIIIII GGGGGGGG NN NN
EEEEEEEEEE DDDDDDDD DDDDDDDD EEEEEEEEE SSSSSSSS IIIIII GGGGGGGG NN NN
EE DD DD FF DD DD EE SS II GG NN NN
EE DD DD FF DD DD EE SS II GG NN NN
EE DD DD FF DD DD EE SS II GG NN NN
EE DD DD FF DD DD EE SS II GG NN NN
EEEEEEEEEE DD DD FFFFFFFF DD DD EEEEEEEEE SSSSSS II II GGGGGG NN NN
EEEEEEEEEE DD DD FFFFFFFF DD DD EEEEEEEEE SSSSSS II II GGGGGG NN NN
EE DD DD FF DD DD EE SS II II GGGGGG NN NN
EE DD DD FF DD DD EE SS II II GGGGGG NN NN
EE DD DD FF DD DD EE SS II II GGGGGG NN NN
EE DD DD FF DD DD EE SS II II GGGGGG NN NN
EEEEEEEEEE DDDDDDDD FF DD DDDDDDDD EEEEEEEEE SSSSSSSS IIIIII GGGGGG NN NN
EEEEEEEEEE DDDDDDDD FF DD DDDDDDDD EEEEEEEEE SSSSSSSS IIIIII GGGGGG NN NN
...

```

```

0001      [ IDENT ('V04-000'),
0002      { ++
0003      *****
0004      **
0005      **  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0006      **  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0007      **  ALL RIGHTS RESERVED.
0008      **
0009      **  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0010      **  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0011      **  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0012      **  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0013      **  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0014      **  TRANSFERRED.
0015      **
0016      **  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0017      **  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0018      **  CORPORATION.
0019      **
0020      **  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0021      **  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0022      **
0023      **
0024      *****

```

FACILITY: VAX/VMS EDF (EDIT/FDL) UTILITY

ABSTRACT: This facility is used to create, modify, and optimize FDL specification files.

ENVIRONMENT: NATIVE/USER MODE

AUTHOR: Ken F. Henderson Jr.

CREATION DATE: 27-Mar-1981

MODIFIED BY:

V03-011 RRB0009 Rowland R. Bradley 22 Jan 1984
Enhancement for display of # of buckets in index,
pages to cache index, and average # key exams.

V03-010 KFH0010 Ken Henderson 8 Aug 1983
Changes for seperate compilation.

V03-009 KFH0009 Ken Henderson 27 Jul 1983
Fix to CALC_ALLOC to prevent div by 0.
Fixed record and bucket overhead
calculations in prologue3_buckets and
prologue3_depth.

V03-008 KFH0008 Ken Henderson 27 May 1983
Fix insertion of DATA_RECORD COMPRESSION
into database to only do it for Key 0.

0058
0059
0060
0061
0062
0063
0064
0065
0066
0067
0068
0069
0070
0071
0072
0073
0074
0075
0076
0077
0078
0079
0080
0081
0082
0083
0084
0085
0086
0087
0088
0089
0090
0091
0092
0093
0094
0095
0096
0097

V03-007 KFH0007 Ken Henderson 26 Apr 1983
Fix location of breakpoint_right.
Add reset of IDATAEDFSK_Y-LOW/HIGH/INCR]
in SETUP_GRAPH. Clean up calls to
ASK_KEY_SIZE, ASK_KEY_POSITION.
Move call to ASK_GLOBAL_WANTED
to APPEND_DEF.

V03-006 KFH0006 Ken Henderson 14 Apr 1983
Removed mandatory VIEWS after
each design. Changed lib\$wait(5.0)
to (3.0). Took out DESIGN_STYLE.
Consolidated PLOT_SIMPLE_GRAPH
and PLOT_SURFACE_GRAPH. Added
SHUFFLE_AREAS and MERGE_AREA
to implement GRANULARITY.

V03-005 KFH0005 Ken Henderson 20 Jan 1983
Added support for DEPTHPOINTS
and removed references to DASH.

V03-004 KFH0004 Ken Henderson 22 Nov 1982
Combined SURFACE_DESIGN and
LINE_DESIGN into one routine.

V03-003 KFH0003 Ken Henderson 8 Sept 1982
Modified most references to main
variables to fit with database
reorganization.

V03-002 KFH0002 Ken Henderson 23-Mar-1982
Modified several routines to fix FT2
QAR 746

V03-001 KFH0001 Ken Henderson 17-Mar-1982
Modified several routines to fix FT2
QARs 509,559,510,574

-- }

EDF DESIGN
V04-000

Source Listing

16-Sep-1984 01:10:30
5-Sep-1984 13:36:36

VAX-11 Pascal V2.4-277 Page 3
DISK\$VMSMASTER:[EDF.SRC]EDFDESIGN.PAS;1 (2)

```

0099 ENVIRONMENT ('LIB$:EDFDESIGN'),
0100
0101 INHERIT (
0102
0103     'SYSSLIBRARY:STARLET',
0104     'SHRLIB$:FDLPARDEF',
0105     'LIB$:EDFSDLMSG',
0106     'LIB$:EDFSTRUCT',
0107     'LIB$:EDFCONST',
0108     'LIB$:EDFTYPE',
0109     'LIB$:EDFVAR',
0110     'LIB$:EDFEXTERN',
0111     'LIB$:EDFCHF',
0112     'LIB$:EDFUTIL',
0113     'LIB$:EDFASK',
0114     'LIB$:EDFSHOW'
0115 )
0116
0117 MODULE EDFDESIGN (INPUT,OUTPUT):
0118

```

[illegible]

```
0120 { ++
0121
0122 PROLOGUE3_BUCKETS -- Routine to calculate the number of buckets at a level.
0123
0124 This routine combines the various file parameters of a prologue3 file and
0125 calls itself recursively to find the number of buckets at each level.
0126
0127 CALLING SEQUENCE:
0128
0129 PROLOGUE3_BUCKETS (INIT_NUMBER_RECORDS,ADDED_NUMBER_RECORDS,INDEX_LEVEL);
0130
0131 INPUT PARAMETERS:
0132
0133 NUMBER_RECORDS
0134 INDEX_LEVEL
0135
0136 IMPLICIT INPUTS:
0137
0138 VARIABLE RECORDS
0139 BDATA[EDF$K_KEY_DUPS]
0140 IDATA[EDF$K_KEY_SIZE]
0141 IDATA[EDF$K_MEAN_RECORD_SIZE]
0142 RDATA[EDF$K_LOAD_FILL]
0143 BYTES_PER_BUCKET
0144
0145 OUTPUT PARAMETERS:
0146
0147 none
0148
0149 IMPLICIT OUTPUTS:
0150
0151 INIT_NUMBER_BUCKETS
0152 ADDED_NUMBER_BUCKETS
0153 DEEPEST
0154
0155 ROUTINES CALLED:
0156
0157 PROLOGUE3_BUCKETS
0158 LIB$SIGNAL
0159
0160 ROUTINE VALUE:
0161
0162 none
0163
0164 SIGNALS:
0165
0166 EDF$_CTRLZ - if a file > 31 index levels was spec'd
0167
0168 SIDE EFFECTS:
0169
0170 none
0171
0172 -- }
```



```
0174 PROCEDURE PROLOGUE3_BUCKETS (
0175     INIT_NUMBER_RECORDS      : INTEGER;
0176     ADDED_NUMBER_RECORDS     : INTEGER;
0177     INDEX_LEVEL              : INTEGER;
0178 );
0179
0180 VAR
0181     INIT_RECORDS_PER_BUCKET   : INTEGER;
0182     ADDED_RECORDS_PER_BUCKET  : INTEGER;
0183     RECORD_OVERHEAD           : INTEGER;
0184     RECORD_SIZE               : INTEGER;
0185     INIT_AVAILABLE_BYTES      : INTEGER;
0186     ADDED_AVAILABLE_BYTES     : INTEGER;
0187     KEY_SAVINGS               : INTEGER;
0188     DATA_SAVINGS             : INTEGER;
0189     INDEX_SAVINGS             : INTEGER;
0190     BUCKET_OVERHEAD           : INTEGER;
0191     TEMP_REC                  : INTEGER;
0192     FOUND                     : BOOLEAN;
0193
0194 BEGIN
0195     ( +
0196     Level 0 is the data level, calculate the filling of the data buckets.
0197     - )
0198     BUCKET_OVERHEAD := CALC_BUC_OVERHEAD(INDEX_LEVEL);
0199     RECORD_OVERHEAD := CALC_REC_OVERHEAD(INDEX_LEVEL);
0200
0201     IF INDEX_LEVEL = 0 THEN
0202     BEGIN
0203         IF IDATA[EDF$K_ACTIVE_KEY] = 0 THEN
0204         BEGIN
0205             ( +
0206             DATA BUCKET
0207             - )
0208
0209             ( +
0210             Combine the two compression factors to get one to weight the record
0211             size by.
0212             - )
0213             KEY_SAVINGS :=
0214                 TRUNC (IDATA[EDF$K_KEY_SIZE] * RDATA[EDF$K_DATA_KEY_COMP]);
0215             DATA_SAVINGS :=
0216                 TRUNC ((IDATA[EDF$K_MEAN_RECORD_SIZE] - IDATA[EDF$K_KEY_SIZE])
0217                     * RDATA[EDF$K_DATA_RECORD_COMP]);
0218
0219             ( +
0220             The 'actual' record size will have the compression subtracted from it.
0221             - )
0222             RECORD_SIZE :=
0223                 IDATA[EDF$K_MEAN_RECORD_SIZE] - (KEY_SAVINGS + DATA_SAVINGS);
0224
0225         END ( IF TRUE KEY = 0 )
0226     END
0227
0228
0229
0230
```

```
0231 ELSE
0232 BEGIN
0233   { +
0234   SIDR BUCKET
0235   - }
0236
0237   INDEX_SAVINGS :=
0238     TRUNC (IDATA[EDF$K_KEY_SIZE] * RDATA[EDF$K_DATA_KEY_COMP]);
0239
0240   TEMP_REC := IDATA[EDF$K_KEY_SIZE] - INDEX_SAVINGS;
0241
0242   TEMP_REC := TEMP_REC +
0243     (IDATA[EDF$K_NUMBER_DUPS] * IRC$C_RRVOVHSZ3);
0244
0245   RECORD_SIZE := TEMP_REC DIV (IDATA[EDF$K_NUMBER_DUPS] + 1);
0246
0247   IF (TEMP_REC MOD (IDATA[EDF$K_NUMBER_DUPS] + 1) <> 0) THEN
0248     RECORD_SIZE := RECORD_SIZE + 1;
0249
0250   END; { IF FALSE KEY = 0 }
0251
0252   END { IF TRUE INDEX_LEVEL = 0 (DATA LEVEL) }
0253
0254 ELSE
0255   { +
0256   For the index levels (L>0), the overheads are as follows.
0257   - }
0258   BEGIN
0259     { +
0260     INDEX BUCKET
0261     - }
0262
0263     INDEX_SAVINGS :=
0264       TRUNC (IDATA[EDF$K_KEY_SIZE] * RDATA[EDF$K_INDEX_RECORD_COMP]);
0265
0266     RECORD_SIZE := IDATA[EDF$K_KEY_SIZE] - INDEX_SAVINGS;
0267
0268   END; { IF FALSE INDEX_LEVEL = 0 }
0269
0270   { +
0271   Now that we've figured out the overheads, how many records can we fit
0272   in a bucket at this level?
0273   - }
0274
0275   { +
0276   First figure out how many bytes are available to use for records.
0277   - }
0278
0279   INIT_AVAILABLE_BYTES :=
0280     TRUNC ((BYTES_PER_BUCKET - BUCKET_OVERHEAD) * RDATA[EDF$K_LOAD_FILL]);
0281
0282   ADDED_AVAILABLE_BYTES :=
0283     TRUNC ((BYTES_PER_BUCKET - BUCKET_OVERHEAD) * RDATA[EDF$K_ADDED_FILL]);
0284
0285
0286
0287
```



```
0288 ( +
0289 The number of records that will fit is simply the space available
0290 divided by the space for each record. (integer division)
0291 - )
0292 INIT_RECORDS_PER_BUCKET :=
0293     INIT_AVAILABLE_BYTES DIV (RECORD_SIZE + RECORD_OVERHEAD);
0294 ADDED_RECORDS_PER_BUCKET :=
0295     ADDED_AVAILABLE_BYTES DIV (RECORD_SIZE + RECORD_OVERHEAD);
0296
0297 ( +
0298 CONVERT or RMS will put at least one (1) record in a data level bucket.
0299 And it will put at least two (2) records in an index level bucket.
0300 - )
0301 IF (INDEX_LEVEL = 0) AND (INIT_RECORDS_PER_BUCKET < 1) THEN
0302     INIT_RECORDS_PER_BUCKET := 1
0303 ELSE IF (INDEX_LEVEL > 0) AND (INIT_RECORDS_PER_BUCKET < 2) THEN
0304     INIT_RECORDS_PER_BUCKET := 2;
0305 IF (INDEX_LEVEL = 0) AND (ADDED_RECORDS_PER_BUCKET < 1) THEN
0306     ADDED_RECORDS_PER_BUCKET := 1
0307 ELSE IF (INDEX_LEVEL > 0) AND (ADDED_RECORDS_PER_BUCKET < 2) THEN
0308     ADDED_RECORDS_PER_BUCKET := 2;
0309 ( + Record the number of buckets for later.
0310 - )
0311 RECS_PER_BUCKET [INDEX_LEVEL] :=
0312     INIT_RECORDS_PER_BUCKET + ADDED_RECORDS_PER_BUCKET;
0313
0314 ( +
0315 Now record the number of buckets at this level.
0316 - )
0317 INIT_NUMBER_BUCKETS [INDEX_LEVEL] :=
0318     INIT_NUMBER_RECORDS DIV INIT_RECORDS_PER_BUCKET;
0319 ADDED_NUMBER_BUCKETS [INDEX_LEVEL] :=
0320     ADDED_NUMBER_RECORDS DIV ADDED_RECORDS_PER_BUCKET;
0321
0322 ( +
0323 If there was a remainder, we need just one more bucket at this level.
0324 - )
0325 IF (INIT_NUMBER_RECORDS MOD INIT_RECORDS_PER_BUCKET) <> 0 THEN
0326     INIT_NUMBER_BUCKETS [INDEX_LEVEL] :=
0327         INIT_NUMBER_BUCKETS [INDEX_LEVEL] + 1;
0328 IF (ADDED_NUMBER_RECORDS MOD ADDED_RECORDS_PER_BUCKET) <> 0 THEN
0329     ADDED_NUMBER_BUCKETS [INDEX_LEVEL] :=
0330         ADDED_NUMBER_BUCKETS [INDEX_LEVEL] + 1;
0331
0332 ( +
0333 Save the number of buckets for later if this is key 0.
```

```
0345 They are used in global buffer count calculations.
0346 - )
0347 IF IDATA[EDF$K_ACTIVE_KEY] = 0 THEN
0348 BEGIN
0349     INIT_PRIMARY_BUCKETS [INDEX_LEVEL] :=
0350     INIT_NUMBER_BUCKETS [INDEX_LEVEL];
0351     ADDED_PRIMARY_BUCKETS [INDEX_LEVEL] :=
0352     ADDED_NUMBER_BUCKETS [INDEX_LEVEL];
0353
0354 END;
0355
0356 { +
0357 Bump the high-water marker.
0358 - )
0359 DEEPEST := INDEX_LEVEL;
0360
0361 { +
0362 If we're at the data level, or we had more than one bucket at this level,
0363 then repeat the calculations for the next level up (down?).
0364 - )
0365 IF (
0366     (INDEX_LEVEL = 0)
0367     OR
0368     (INIT_NUMBER_BUCKETS [INDEX_LEVEL] > 1)
0369     OR
0370     (ADDED_NUMBER_BUCKETS [INDEX_LEVEL] > 1)
0371 ) THEN
0372 BEGIN
0373     { +
0374     In the index, the records merely point to buckets.
0375     - )
0376     IF INDEX_LEVEL = 0 THEN
0377     BEGIN
0378         FOUND := FALSE;
0379         IF OPTIMIZING THEN
0380         BEGIN
0381             POINT_AT_ANALYSIS;
0382             FOUND := FIND_OBJECT (SEC_ANALYSIS_OF_KEY,
0383                                   IDATA[EDF$K_ACTIVE_KEY],
0384                                   LEVEL1_RECORD_COUNT, 0);
0385             POINT_AT_DEFINITION;
0386         END;
0387         IF FOUND THEN
```

```
0402 BEGIN
0403
0404     INIT_NUMBER_RECORDS := DEF_CURRENT^.NUMBER;
0405
0406 END
0407
0408 ELSE
0409
0410 BEGIN
0411
0412     INIT_NUMBER_RECORDS := INIT_NUMBER_BUCKETS [INDEX_LEVEL];
0413
0414 END;
0415
0416 END
0417
0418 ELSE
0419
0420 BEGIN
0421
0422     INIT_NUMBER_RECORDS := INIT_NUMBER_BUCKETS [INDEX_LEVEL];
0423
0424 END;
0425
0426 ADDED_NUMBER_RECORDS := ADDED_NUMBER_BUCKETS [INDEX_LEVEL];
0427
0428 INDEX_LEVEL := INDEX_LEVEL + 1;
0429
0430 ( +
0431 Pathological file here - tell the user and pop him up.
0432 - )
0433 IF INDEX_LEVEL > 31 THEN
0434
0435 BEGIN
0436
0437     WRITELN (SHIFT,ANSI_REVERSE,
0438 ' A File of Greater than 31 Index Levels has been specified. ',
0439 ANSI_RESET);
0440
0441     LIB$WAIT (3.0);
0442
0443     LIB$SIGNAL (EDF$_CTRLZ,0,0,0);
0444
0445 END;
0446
0447 ( +
0448 Recurse to the next level.
0449 - )
0450 PROLOGUE3_BUCKETS (
0451     INIT_NUMBER_RECORDS,
0452     ADDED_NUMBER_RECORDS,
0453     INDEX_LEVEL
0454 );
0455
0456 END;
0457
0458 END; ( PROLOGUE3_BUCKETS )
```



```
0460 { **
0461
0462 PROLOGUE3_DEPTH -- Routine to calculate the depth of a prologue3 index.
0463
0464 This routine combines the various file parameters of a prologue3 file and
0465 'builds' and index from the data level up to the root - to find its depth.
0466
0467 CALLING SEQUENCE:
0468
0469 DEPTH := PROLOGUE3_DEPTH;
0470
0471 INPUT PARAMETERS:
0472
0473 none
0474
0475 IMPLICIT INPUTS:
0476
0477 TOTAL RECORDS
0478 IDATA[EDF$K_BLOCKS_IN_BUCKET]
0479 DEEPEST
0480
0481 OUTPUT PARAMETERS:
0482
0483 none
0484
0485 IMPLICIT OUTPUTS:
0486
0487 BYTES PER BUCKET
0488 NUMBER_BUCKETS
0489
0490 ROUTINES CALLED:
0491
0492 PROLOGUE3_BUCKETS
0493
0494 ROUTINE VALUE:
0495
0496 Depth of the index
0497
0498 SIGNALS:
0499
0500 none
0501
0502 SIDE EFFECTS:
0503
0504 none
0505
0506 -- }
```

```
0508 FUNCTION PROLOGUE3_DEPTH : INTEGER;
0509
0510 VAR
0511     BUCKET_OVERHEAD : INTEGER;
0512     RECORD_OVERHEAD : INTEGER;
0513     RECORD_SIZE      : INTEGER;
0514     I                 : INTEGER;
0515
0516 BEGIN
0517
0518     { +
0519     Clear out the arrays that holds the number of buckets per level.
0520     - }
0521     FOR I := 0 TO 31 DO
0522
0523     BEGIN
0524
0525         INIT_NUMBER_BUCKETS [I]      := 0;
0526         ADDED_NUMBER_BUCKETS [I]      := 0;
0527         RECS_PER_BUCRET [I]          := 0;
0528
0529     END;
0530
0531     { +
0532     Convert block/bucket to bytes/bucket.
0533     - }
0534     BYTES_PER_BUCKET      := IDATA[EDF$K_BLOCKS_IN_BUCKET] * 512;
0535
0536     { +
0537     Reset depth and calculate how deep the index will be.
0538     - }
0539     DEEPEST               := 0;
0540
0541     { +
0542     Figure depth only if the record will fit in the bucket.
0543     Otherwise flag it.
0544     - }
0545     BUCKET_OVERHEAD       := CALC_BUC_OVERHEAD(0);
0546     RECORD_OVERHEAD       := CALC_REC_OVERHEAD(0);
0547
0548     IF IDATA[EDF$K_MAX_RECORD_SIZE] = 0 THEN
0549         RECORD_SIZE       := CUR_MAX_REC
0550
0551     ELSE
0552
0553         RECORD_SIZE       := IDATA[EDF$K_MAX_RECORD_SIZE];
0554
0555     { +
0556     Only do the depth calculation if the record will fit in the bucket,
0557     and the key will fit in the record.
0558     - }
0559     IF (
0560         ((BYTES_PER_BUCKET - (BUCKET_OVERHEAD + RECORD_OVERHEAD)) >=
0561         IDATA[EDF$K_MEAN_RECORD_SIZE])
0562         AND
0563         (RECORD_SIZE >= (IDATA[EDF$K_KEY_SIZE] + IDATA[EDF$K_KEY_POSITION]))
0564
```

```
0565      ) THEN
0566
0567 BEGIN
0568
0569     CASE IDATA[EDFSK_LOAD_METHOD] OF
0570
0571     EDFSK_FAST_CONVERT :
0572         RDATA[EDFSK_LOAD_FILL] := IDATA[EDFSK_DESIRED_FILL] / 100.0;
0573
0574     EDFSK_NOFAST_CONVERT :
0575         IF BDATA[EDFSK_ASCENDING_LOAD] THEN
0576             RDATA[EDFSK_LOAD_FILL] :=
0577                 0.90 * (IDATA[EDFSK_DESIRED_FILL] / 100.0)
0578
0579         ELSE
0580             RDATA[EDFSK_LOAD_FILL] :=
0581                 0.6667 * (IDATA[EDFSK_DESIRED_FILL] / 100.0);
0582
0583     EDFSK_RMS_PUTS :
0584 BEGIN
0585     IF BDATA[EDFSK_ASCENDING_LOAD] THEN
0586         RDATA[EDFSK_LOAD_FILL] :=
0587             0.90 * (IDATA[EDFSK_DESIRED_FILL] / 100.0)
0588
0589     ELSE
0590         RDATA[EDFSK_LOAD_FILL] :=
0591             0.6667 * (IDATA[EDFSK_DESIRED_FILL] / 100.0);
0592
0593     IDATA[EDFSK_FDL_FILL] := 100;
0594
0595 END;
0596 OTHERWISE
0597     { NULL-STATEMENT } ;
0598
0599 END;    { CASE }
0600
0601 IF BDATA[EDFSK_ASCENDING_ADDED] THEN
0602     RDATA[EDFSK_ADDED_FILL] := 0.90
0603
0604 ELSE
0605     RDATA[EDFSK_ADDED_FILL] := 0.6667;
0606
0607 PROLOGUE3_BUCKETS(IDATA[EDFSK_INITIAL_COUNT], IDATA[EDFSK_ADDED_COUNT], 0);
0608
0609 { +
0610
```


EDFDESIGN
V04-000

Source Listing

M 4
16-Sep-1984 01:10:30
5-Sep-1984 13:36:36

VAX-11 Pascal V2.4-277
DISK\$VMSMASTER:[EDF.SRC]EDFDESIGN.PAS;1 (6) Page 13

```
0622       The deepest we went is the function value.  
0623       - }  
0624       PROLOGUE3_DEPTH      := DEEPEST;  
0625  
0626       END  
0627  
0628       ELSE  
0629  
0630       PROLOGUE3_DEPTH      := 0;  
0631  
0632     END; { PROLOGUE3_DEPTH }
```

```
0634      ( ++
0635
0636      NATURAL_DEPTH -- Find most typical depth of file.
0637
0638      This routine does calculations to find out the most reasonable bucket size
0639      for an index.
0640
0641      CALLING SEQUENCE:
0642
0643      BUCKET_DEFAULT := NATURAL_DEPTH;
0644
0645      INPUT PARAMETERS:
0646
0647      none
0648
0649      IMPLICIT INPUTS:
0650
0651      none
0652
0653      OUTPUT PARAMETERS:
0654
0655      none
0656
0657      IMPLICIT OUTPUTS:
0658
0659      COLOR_ROW
0660
0661      ROUTINES CALLED:
0662
0663      none
0664
0665      ROUTINE VALUE:
0666
0667      BUCKET_DEFAULT
0668
0669      SIGNALS:
0670
0671      none
0672
0673      SIDE EFFECTS:
0674
0675      none
0676
0677      -- }
```

```
0679 [GLOBAL] FUNCTION NATURAL_DEPTH : INTEGER;
0680
0681 VAR
0682     DEPTH          : ARRAY [1..BKTSC_MAXBKTSIZ] OF INTEGER;
0683     TALLY          : ARRAY [1..BKTSC_MAXBKTSIZ] OF REAL;
0684     CURRENT_WEIGHT : REAL;
0685     CURRENT_TALLY  : REAL;
0686     MAX_TALLY      : REAL;
0687     TEMP_DIST      : INTEGER;
0688     LEFT_ADJ_RANGE : INTEGER;
0689     CURRENT_DEPTH  : INTEGER;
0690     RANGE          : INTEGER;
0691     MAX_RANGE      : INTEGER;
0692     MIN_BKS        : INTEGER;
0693
0694
0695
0696     PROCEDURE EXTEND_INDEX_INFO (VAR EXAMPOINT,
0697                                   NUMPOINT,
0698                                   PAGEPOINT,
0699                                   BREAKPOINT : INTEGER);
0700
0701     { +
0702       Calculate and save more index information.
0703     - }
0704     VAR
0705         I          : INTEGER;
0706
0707     BEGIN
0708         IDATA [EDFSK_BLOCKS_IN_BUCKET] := BREAKPOINT;
0709         TEMP_DIST := PROLOGUE3_DEPTH;
0710         EXAMPOINT := 0;
0711         NUMPOINT  := 0;
0712
0713         FOR I := 1 TO 31 DO
0714             BEGIN
0715                 EXAMPOINT := EXAMPOINT + RECS_PER_BUCKET [I];
0716                 NUMPOINT  :=
0717                     NUMPOINT + INIT_NUMBER_BUCKETS [I] + ADDED_NUMBER_BUCKETS [I];
0718
0719             END; { FOR }
0720
0721         EXAMPOINT := EXAMPOINT DIV 2;
0722         PAGEPOINT := NUMPOINT * BREAKPOINT;
0723
0724     END; { procedure EXTEND_INDEX_INFO }
0725
0726
0727     { +
0728     Main Function Begins Here
0729     - }
0730     BEGIN
0731
0732         BREAKPOINT_RIGHT := 0;
0733
0734         { +
0735         Fill the depth array with the depths at each bucketsize.
```



```
0736 And zero out the tally array.  
0737 - )  
0738 FOR RANGE := 1 TO BKT$C_MAXBKTSIZ DO  
0739  
0740 BEGIN  
0741     IDATA[EDF$K_BLOCKS_IN_BUCKET] := RANGE;  
0742     DEPTH[RANGE] := PROLOGUE3_DEPTH;  
0743     TALLY[RANGE] := 0;  
0744  
0745  
0746 END; { FOR }  
0747  
0748 { +  
0749 Add up the lengths of the ranges.  
0750 - }  
0751 CURRENT_WEIGHT := 1.0;  
0752 CURRENT_DEPTH := 0;  
0753 CURRENT_TALLY := 0;  
0754  
0755 FOR RANGE := BKT$C_MAXBKTSIZ DOWNT0 1 DO  
0756  
0757 BEGIN  
0758     IF DEPTH[RANGE] = 0 THEN  
0759  
0760     BEGIN  
0761  
0762     IF RANGE < BKT$C_MAXBKTSIZ THEN  
0763  
0764     IF DEPTH[RANGE+1] > 0 THEN  
0765  
0766     TALLY[RANGE+1] := CURRENT_TALLY;  
0767  
0768     TALLY[RANGE] := 0;  
0769  
0770     END  
0771  
0772     ELSE IF DEPTH[RANGE] > CURRENT_DEPTH THEN  
0773  
0774     BEGIN  
0775  
0776     IF RANGE < BKT$C_MAXBKTSIZ THEN  
0777  
0778     TALLY[RANGE+1] := CURRENT_TALLY;  
0779  
0780     CURRENT_DEPTH := DEPTH[RANGE];  
0781     CURRENT_TALLY := CURRENT_WEIGHT;  
0782  
0783     END  
0784  
0785     ELSE  
0786  
0787     BEGIN  
0788  
0789     { +  
0790 Bucket sizes from 33 to 63 aren't added in.  
0791 - }  
0792
```

```
0793       IF RANGE < 33 THEN
0794
0795           CURRENT_TALLY := CURRENT_TALLY + CURRENT_WEIGHT;
0796
0797       END;
0798
0799       IF IDATA[EDFSK_BUCKET_WEIGHT] = EDFSK_SMALLER_BUFFERS THEN
0800
0801           CURRENT_WEIGHT := CURRENT_WEIGHT + BUCKET_LEFT_WEIGHT;
0802
0803       END;      { FOR }
0804
0805       MAX_TALLY      := 0;
0806       MAX_RANGE      := 0;
0807       MIN_BKS        := 1;
0808
0809       { +
0810       Minimum bucket size may be greater than one. Determine it here.
0811       - }
0812       FOR RANGE := 1 TO BKTSC_MAXBKTSIZ DO
0813
0814           IF DEPTH[RANGE] < 1 THEN
0815               BEGIN
0816                   MIN_BKS := RANGE + 1;
0817               END;
0818
0819       { +
0820       Now find the left end of the most common range (that's not 0).
0821       - }
0822       FOR RANGE := BKTSC_MAXBKTSIZ DOWNT0 MIN_BKS DO
0823
0824           IF TALLY[RANGE] > MAX_TALLY THEN
0825
0826               BEGIN
0827
0828                   MAX_TALLY := TALLY[RANGE];
0829                   MAX_RANGE := RANGE;
0830
0831               END;
0832
0833       { +
0834       Sometimes there aren't any values at all on a row...
0835       - }
0836       IF MAX_RANGE < 1 THEN
0837
0838           MAX_RANGE := 1;
0839
0840       { +
0841       Now let's calculate what the colors are for this row.
0842       Right part 1st...
0843       - }
0844       FOR RANGE := MAX_RANGE TO BKTSC_MAXBKTSIZ DO
0845
0846       BEGIN
0847
0848           TEMP_DIST := RANGE - MAX_RANGE;
0849
```

```
0850 IF TEMP_DIST < 9 THEN
0851 BEGIN
0852     COLOR_ROW[RANGE-1] := EDF$C_LIGHT_GREEN;
0853 END
0854 ELSE IF (
0855     (TEMP_DIST > 8)
0856     AND
0857     (TEMP_DIST < 21)
0858 ) THEN
0859 BEGIN
0860     COLOR_ROW[RANGE-1] := EDF$C_MEDIUM_YELLOW;
0861 END
0862 ELSE
0863 BEGIN
0864     COLOR_ROW[RANGE-1] := EDF$C_DARK_RED;
0865 END;
0866 { +
0867 Make sure the green region includes only one depth.
0868 - }
0869 IF (
0870     (DEPTH[RANGE] <> DEPTH[MAX_RANGE])
0871     AND
0872     (COLOR_ROW[RANGE-1] = EDF$C_LIGHT_GREEN)
0873 ) THEN
0874     COLOR_ROW[RANGE-1] := EDF$C_MEDIUM_YELLOW;
0875 { +
0876 If there's a point where we can get even a flatter file,
0877 note that.
0878 - }
0879 IF (
0880     (DEPTH[RANGE] < DEPTH[MAX_RANGE])
0881     AND
0882     (BREAKPOINT_RIGHT = 0)
0883 ) THEN
0884     BREAKPOINT_RIGHT := MAX_FACTOR (IDATA[EDF$K_CLUSTER_SIZE],
0885                                     RANGE, BKT$C_MAXBKT$IZ);
0886 END; { FOR }
0887 { +
0888 Now do to the left of natural.
0889 - }
```



```
0907 IF MAX_RANGE = 1 THEN
0908
0909 BEGIN
0910
0911     COLOR_ROW[0] := EDF$C_LIGHT_GREEN;
0912     LEFT_ADJ_RANGE := DEPTH[MAX_RANGE];
0913
0914 END
0915
0916 ELSE
0917
0918 BEGIN
0919
0920     LEFT_ADJ_RANGE := DEPTH[MAX_RANGE-1];
0921
0922     FOR RANGE := (MAX_RANGE-1) DOWNT0 1 DO
0923
0924     BEGIN
0925
0926         IF DEPTH[RANGE] = LEFT_ADJ_RANGE THEN
0927
0928             COLOR_ROW[RANGE-1] := EDF$C_MEDIUM_YELLOW
0929
0930         ELSE
0931
0932             COLOR_ROW[RANGE-1] := EDF$C_DARK_RED;
0933
0934     END;
0935
0936 END;      { IF FALSE MAX_RANGE = 1 }
0937
0938 { +
0939 Now blank out any illegal spots.
0940 - }
0941 FOR RANGE := 1 TO BKT$C_MAXBKTSIZ DO
0942
0943     IF DEPTH[RANGE] < 1 THEN
0944     BEGIN
0945         COLOR_ROW[RANGE-1] := EDF$C_BACKGROUND_COLOR;
0946     END;
0947
0948 { +
0949 Now fill in the breakpoint variables.
0950 Mid is easy.
0951 - }
0951 BREAKPOINT_MID := MAX_FACTOR (IDATA[EDF$K_CLUSTER_SIZE],
0952                                MAX_RANGE,BKT$C_MAXBKTSIZ);
0953
0954 IF BREAKPOINT_RIGHT = 0 THEN
0955
0956 BEGIN
0957
0958     { +
0959 Breakpoint_right.
0960 - }
0961     RANGE := MAX_RANGE;
0962
0963     WHILE (
```

```
0964 (RANGE < BKTSC_MAXBKTSIZ)
0965 AND
0966 (COLOR_ROW[RANGE-1] = EDFSC_LIGHT_GREEN)
0967 ) DO
0968
0969     RANGE := RANGE + 1;
0970
0971 IF COLOR_ROW[RANGE-1] <> EDFSC_BACKGROUND_COLOR THEN
0972
0973     BREAKPOINT_RIGHT := MAX_FACTOR (IDATA[EDFSK_CLUSTER_SIZE],
0974                                     RANGE, BKTSC_MAXBKTSIZ)
0975
0976 ELSE IF RANGE <> MAX_RANGE THEN
0977
0978     BREAKPOINT_RIGHT := MAX_FACTOR (IDATA[EDFSK_CLUSTER_SIZE],
0979                                     (RANGE-1), BKTSC_MAXBKTSIZ)
0980
0981 ELSE
0982
0983     BREAKPOINT_RIGHT := MAX_FACTOR (IDATA[EDFSK_CLUSTER_SIZE],
0984                                     MAX_RANGE, BKTSC_MAXBKTSIZ);
0985
0986 END; ( IF BREAKPOINT_RIGHT = 0 )
0987
0988 { +
0989 Breakpoint_Left.
0990 - }
0991 RANGE := MAX_RANGE - 1;
0992
0993 IF RANGE > 0 THEN
0994
0995     WHILE (RANGE > 1) AND (DEPTH[RANGE] = LEFT_ADJ_RANGE) DO
0996
0997         RANGE := RANGE - 1;
0998
0999 { +
1000 Backup
1001 - }
1002 RANGE := RANGE + 1;
1003
1004 IF RANGE >= MAX_RANGE THEN
1005
1006     BREAKPOINT_LEFT := MAX_FACTOR (IDATA[EDFSK_CLUSTER_SIZE],
1007                                     MAX_RANGE, BKTSC_MAXBKTSIZ)
1008
1009 ELSE
1010
1011     BREAKPOINT_LEFT := MAX_FACTOR (IDATA[EDFSK_CLUSTER_SIZE],
1012                                     RANGE, BKTSC_MAXBKTSIZ);
1013
1014 { +
1015 Now stuff the depthpoint variables.
1016 - }
1017 DEPTHPOINT_LEFT := DEPTH[BREAKPOINT_LEFT];
1018 DEPTHPOINT_MID := DEPTH[BREAKPOINT_MID];
1019 DEPTHPOINT_RIGHT := DEPTH[BREAKPOINT_RIGHT];
1020
```

```
1021 { +
1022 Calculate and save more index information. Left side display.
1023 - }
1024 EXTEND_INDEX_INFO ( EXAMPOINT_LEFT, NUMPOINT_LEFT,
1025                     PAGEPOINT_LEFT, BREAKPOINT_LEFT);
1026
1027 { +
1028 Calculate and save more index information. Mid of display.
1029 - }
1030 EXTEND_INDEX_INFO ( EXAMPOINT_MID, NUMPOINT_MID,
1031                     PAGEPOINT_MID, BREAKPOINT_MID);
1032
1033 { +
1034 Calculate and save more index information. Right side display.
1035 - }
1036 EXTEND_INDEX_INFO ( EXAMPOINT_RIGHT, NUMPOINT_RIGHT,
1037                     PAGEPOINT_RIGHT, BREAKPOINT_RIGHT);
1038
1039 { +
1040 Now stuff the function value and leave.
1041 - }
1042 NATURAL_DEPTH      := BREAKPOINT_MID;
1043
1044 END; ( NATURAL_DEPTH )
```

```
1046      ( ++
1047
1048      PLOT_GRAPH -- Calculate index depths and plot them.
1049
1050      This routine figures out what the index depths will be for all bucketsizes
1051      and plots them on the screen.
1052
1053      CALLING SEQUENCE:
1054
1055      PLOT_GRAPH;
1056
1057      INPUT PARAMETERS:
1058
1059      none
1060
1061      IMPLICIT INPUTS:
1062
1063      FIRST_PLOT
1064
1065      OUTPUT PARAMETERS:
1066
1067      none
1068
1069      IMPLICIT OUTPUTS:
1070
1071      SYS$OUTPUT:
1072      IDATA[EDF$K_BLOCKS_IN_BUCKET]
1073      XY_ARRAY
1074
1075      ROUTINES CALLED:
1076
1077      PROLOGUE3_DEPTH
1078      EDF$GRAPH
1079
1080      ROUTINE VALUE:
1081
1082      none
1083
1084      SIGNALS:
1085
1086      none
1087
1088      SIDE EFFECTS:
1089
1090      none
1091
1092      -- }
```



```
1094 PROCEDURE PLOT_GRAPH;
1095
1096 VAR
1097     RANGE                : INTEGER;
1098     GRAPH_SWITCH         : INTEGER;
1099     TEMP_INTEGER         : INTEGER;
1100     TEMP_INT2            : INTEGER;
1101
1102 BEGIN
1103     IF IDATA[EDF$K_SURFACE_OPTION] = EDF$K_LINE_SURFACE THEN
1104     BEGIN
1105         { +
1106         Do the simple graph.
1107         - }
1108         GRAPH_TYPE        := EDF$K_LINE;
1109         Y_LABEL            := 'Index Depth';
1110
1111         { +
1112         Swap the graph_index (for double buffering)
1113         - }
1114         TEMP_INTEGER      := CURRENT_GRAPH_INDEX;
1115         CURRENT_GRAPH_INDEX := LAST_GRAPH_INDEX;
1116         LAST_GRAPH_INDEX  := TEMP_INTEGER;
1117
1118     END;
1119
1120     IF FIRST_PLOT THEN
1121     BEGIN
1122         { +
1123         Hard set the graph index if this is the 1st time through.
1124         Plus set the graph switch to non-move-mode to plot the entire
1125         axis as well as the points.
1126         - }
1127         GRAPH_SWITCH      := -1;
1128         CURRENT_GRAPH_INDEX := 0;
1129         LAST_GRAPH_INDEX  := 1;
1130
1131     END { IF TRUE FIRST_PLOT }
1132
1133     ELSE
1134     BEGIN
1135         { +
1136         Not the 1st time through, just 'move' the points from their
1137         last position.
1138         - }
1139         IF IDATA[EDF$K_SURFACE_OPTION] = EDF$K_LINE_SURFACE THEN
1140             GRAPH_SWITCH      := LAST_GRAPH_INDEX
1141
1142         ELSE
1143             GRAPH_SWITCH      := 1;
1144
1145     END;
1146
1147 END;
```

```
1151 IF IDATA[EDF$K_SURFACE_OPTION] <> EDF$K_LINE_SURFACE THEN
1152
1153 BEGIN
1154     CURRENT_GRAPH_INDEX      := 0;
1155
1156 END
1157
1158 ELSE
1159 BEGIN
1160     { +
1161     Fill the row in the xy_plot with the depths at each bucketsize.
1162     - }
1163     FOR RANGE := 0 TO 31 DO
1164     BEGIN
1165         IDATA[EDF$K_BLOCKS_IN_BUCKET] := RANGE + 1;
1166         XY_PLOT[CURRENT_GRAPH_INDEX,RANGE] := PROLOGUE3_DEPTH;
1167     END; { FOR }
1168
1169     { +
1170     Fill the color_row, and copy that into the array.
1171     - }
1172     TEMP_INTEGER := NATURAL_DEPTH;
1173     FOR TEMP_INT2 := 0 TO 31 DO
1174         COLOR_PLOT[CURRENT_GRAPH_INDEX,TEMP_INT2] := COLOR_ROW[TEMP_INT2];
1175     END; { IF FALSE IDATA[EDF$K_SURFACE_OPTION] <> EDF$K_LINE_SURFACE }
1176
1177 IF NOT AUTO_TUNE THEN
1178 BEGIN
1179     { +
1180     Since edfgrf doesn't for VT125s...
1181     - }
1182     IF REGIS THEN
1183     BEGIN
1184         { +
1185         Force the screen out of reverse video to let all the
1186         characters be visible (VT125 HACK!!!)
1187         - }
1188         WRITELN (''(27)'[?5L)');
1189
1190         { +
1191         Can't use CLEAR (SCREEN) because that also does a graphics clear.
1192         - }
1193         LIB$ERASE_PAGE (LINE_ONE,COL_ONE);
```

```
1208
1209     END;
1210
1211     { +
1212     Plot that graph, tote that barge, lift that bale...
1213     - }
1214     EDF$GRAPH (
1215         GRAPH_TYPE,
1216         XY_PLOT,
1217         CURRENT_GRAPH_INDEX,
1218         GRAPH_SWITCH,
1219         IDATA[EDFSK-Y-HIGH],
1220         IDATA[EDFSK-Y-LOW],
1221         IDATA[EDFSK-Y-INCR],
1222         Y_LABEL,
1223         COLOR_PLOT
1224     );
1225
1226     END;      { IF NOT AUTO_TUNE }
1227
1228     { +
1229     Only DEC CRTs can scroll only at the bottom, so if we don't have one of
1230     those, always do a complete screen rewrite (in case of full screen scroll).
1231     - }
1232     IF DEC_CRT THEN
1233         FIRST_PLOT      := FALSE;
1234
1235     END;      { PLOT_GRAPH }
```

```
1238      ( ++
1239
1240      WARN_OF_ERASE -- Tell user we're about to clobber his definition.
1241
1242      This routine warns the user that we're about to erase the definition and
1243      asks for confirmation.
1244
1245      CALLING SEQUENCE:
1246
1247      WARN_OF_ERASE:
1248
1249      INPUT PARAMETERS:
1250
1251      none
1252
1253      IMPLICIT INPUTS:
1254
1255      SYSS$INPUT:
1256
1257      OUTPUT PARAMETERS:
1258
1259      none
1260
1261      IMPLICIT OUTPUTS:
1262
1263      none
1264
1265      ROUTINES CALLED:
1266
1267      none
1268
1269      ROUTINE VALUE:
1270
1271      none
1272
1273      SIGNALS:
1274
1275      none
1276
1277      SIDE EFFECTS:
1278
1279      none
1280
1281      -- )
```



```
1283 PROCEDURE WARN_OF_ERASE;  
1284  
1285 BEGIN  
1286  
1287     IF NOT AUTO_TUNE THEN  
1288  
1289         BEGIN  
1290  
1291             { +  
1292             If the list has more than the IDENT in it,  
1293             query the user about replacing it.  
1294             - }  
1295             IF (  
1296                 (DEF_HEAD <> DEF_TAIL)  
1297             OR  
1298                 (DEF_HEAD^.PRIMARY <> IDENT)  
1299             ) THEN  
1300  
1301                 BEGIN  
1302  
1303                     IF (  
1304                         ((IDATA[EDFSK_SCRIPT_OPTION] = EDFSK_REDESIGN_FDL)  
1305                         OR  
1306                         (IDATA[EDFSK_SCRIPT_OPTION] = EDFSK_OPTIMIZE_FDL))  
1307                     AND  
1308                         (ISAM ORG)  
1309                     ) THEN  
1310  
1311                         WRITE (SHIFT,ANSI_REVERSE,  
1312                             'The Definition of Key',IDATA[EDFSK_ACTIVE_KEY]:3,  
1313                             ' will be replaced.',CRLF)  
1314  
1315                     ELSE  
1316  
1317                         WRITELN (SHIFT,ANSI_REVERSE,  
1318                             ' The Current Definition will be replaced. ',  
1319                             ANSI_RESET,CRLF);  
1320  
1321                         QUERY (EDFSK_RETURN);  
1322  
1323                     END;      { IF TRUE DEF_HEAD <> DEF_TAIL }  
1324  
1325                 END;      { IF NOT AUTO_TUNE }  
1326  
1327     END;      { WARN_OF_ERASE }
```

```
1329      ( ++
1330
1331      NON_KEY_DEF -- Put into the definition the File, Record, etc stuff.
1332
1333      This routine handles the initial addition of the non-repeating attributes.
1334
1335      CALLING SEQUENCE:
1336
1337      NON_KEY_DEF;
1338
1339      INPUT PARAMETERS:
1340
1341      none
1342
1343      IMPLICIT INPUTS:
1344
1345      none
1346
1347      OUTPUT PARAMETERS:
1348
1349      none
1350
1351      IMPLICIT OUTPUTS:
1352
1353      DEF_CURRENT
1354      DEF_HEAD
1355
1356      ROUTINES CALLED:
1357
1358      none
1359
1360      ROUTINE VALUE:
1361
1362      none
1363
1364      SIGNALS:
1365
1366      none
1367
1368      SIDE EFFECTS:
1369
1370      none
1371
1372      -- }
```

```
1374 PROCEDURE NON_KEY_DEF;  
1375  
1376 BEGIN  
1377  
1378   { +  
1379   Get the rest of the non-key data.  
1380   - }  
1381   QUERY (EDFSK_FDL_TITLE);  
1382   QUERY (EDFSK_DATA_FILE_NAME);  
1383   QUERY (EDFSK_CARR_CTRL);  
1384  
1385   { +  
1386   Now make up the rest of the definition.  
1387   - }  
1388   IF BDATA[EDFSK_FDL_TITLE] THEN  
1389  
1390   BEGIN  
1391  
1392     MAKE_SCRATCH;  
1393  
1394     WITH DEF_SCRATCH^ DO  
1395  
1396     BEGIN  
1397  
1398       { +  
1399       TITLE primary.  
1400       - }  
1401       LIB$COPY_DXDX (SDATA[EDFSK_FDL_TITLE],STRING);  
1402       STR$FREE1_DX (SDATA[EDFSK_FDL_TITLE]);  
1403  
1404       PRIMARY           := TITLE;  
1405       OBJECT_TYPE      := PRI;  
1406  
1407       INSERT_IN_ORDER (REPLACE_OBJ);  
1408  
1409     END; { WITH DEF_SCRATCH^ DO }  
1410  
1411   END { IF TRUE BDATA[EDFSK_FDL_TITLE] }  
1412  
1413   ELSE  
1414  
1415   BEGIN  
1416  
1417     IF FIND_OBJECT (PRI,TITLE,0,DUMMY_SECONDARY$,0) THEN  
1418  
1419       DELETE_CURRENT;  
1420  
1421   END; { IF FALSE BDATA[EDFSK_FDL_TITLE] }  
1422  
1423   MAKE_SCRATCH;  
1424  
1425   WITH DEF_SCRATCH^ DO  
1426  
1427   BEGIN  
1428  
1429     { +  
1430     SYSTEM primary.
```

```
1431      - )
1432      OBJECT_TYPE      := PRI;
1433      PRIMARY          := SYSTEM;
1434
1435      INSERT_IN_ORDER (REPLACE_OBJ);
1436
1437      END;      ( WITH DEF_SCRATCH^ DO )
1438
1439      MAKE_SCRATCH;
1440
1441      WITH DEF_SCRATCH^ DO
1442
1443      BEGIN
1444
1445      ( +
1446      SOURCE Secondary.
1447      - )
1448      PRIMARY          := SYSTEM;
1449      SECONDARY        := SOURCE;
1450      QUALIFIER        := FDL$C_VMS;
1451
1452      INSERT_IN_ORDER (REPLACE_OBJ);
1453
1454      END;      ( WITH DEF_SCRATCH^ DO )
1455
1456      MAKE_SCRATCH;
1457
1458      WITH DEF_SCRATCH^ DO
1459
1460      BEGIN
1461
1462      ( +
1463      FILE primary.
1464      - )
1465      OBJECT_TYPE      := PRI;
1466      PRIMARY          := FILES;
1467
1468      INSERT_IN_ORDER (REPLACE_OBJ);
1469
1470      END;      ( WITH DEF_SCRATCH^ DO )
1471
1472      ( +
1473      NAME secondary.
1474      - )
1475      IF BDATA[EDFSK_DATA_FILE_NAME] THEN
1476
1477      BEGIN
1478
1479      MAKE_SCRATCH;
1480
1481      WITH DEF_SCRATCH^ DO
1482
1483      BEGIN
1484
1485      LIB$COPY_DXDX (SDATA[EDFSK_DATA_FILE_NAME],STRING);
1486      STR$FREE1_DX (SDATA[EDFSK_DATA_FILE_NAME]);
1487
```



```
1488      PRIMARY          := FILES;
1489      SECONDARY         := NAME;
1490
1491      INSERT_IN_ORDER (REPLACE_OBJ);
1492
1493      END;      { WITH DEF_SCRATCH^ }
1494
1495      END      { IF TRUE BDATA[EDFSK_DATA_FILE_NAME] }
1496
1497      ELSE
1498
1499      BEGIN
1500
1501      IF FIND_OBJECT (SEC,FILES,0,NAME,0) THEN
1502
1503      DELETE_CURRENT;
1504
1505      END;      { IF FALSE BDATA[EDFSK_DATA_FILE_NAME] }
1506
1507      MAKE_SCRATCH;
1508
1509      WITH DEF_SCRATCH^ DO
1510
1511      BEGIN
1512
1513      { +
1514      ORGANIZATION secondary.
1515      - }
1516      PRIMARY          := FILES;
1517      SECONDARY         := ORGANIZATION;
1518
1519      CASE IDATA[EDFSK_SCRIPT_OPTION] OF
1520
1521      EDFSK_OPTIMIZE_FDL,
1522      EDFSK_REDESIGN_FDL,
1523      EDFSK_IDX_DESIGN_FDL :      QUALIFIER := FDLSC_IDX;
1524      EDFSK_SEQ_DESIGN_FDL :      QUALIFIER := FDLSC_SEQ;
1525      EDFSK_REL_DESIGN_FDL :      QUALIFIER := FDLSC_REL;
1526
1527      OTHERWISE
1528
1529      { NULL-STATEMENT } ;
1530
1531      END;      { CASE }
1532
1533      INSERT_IN_ORDER (REPLACE_OBJ);
1534
1535      END;      { WITH DEF_SCRATCH^ DO }
1536
1537      MAKE_SCRATCH;
1538
1539      WITH DEF_SCRATCH^ DO
1540
1541      BEGIN
1542
1543      { +
1544      RECORD primary.
```

```
1545      - )
1546      OBJECT_TYPE          := PRI;
1547      PRIMARY              := RECORDS;
1548
1549      INSERT_IN_ORDER (REPLACE_OBJ);
1550
1551  END;      ( WITH DEF_SCRATCH^ DO )
1552
1553  IF IDATA[EDF$K_SCRIPT_OPTION] = EDF$K_SEQ_DESIGN_FDL THEN
1554
1555  BEGIN
1556
1557      { +
1558      BLOCK_SPAN secondary.
1559      - )
1560      MAKE_SCRATCH;
1561
1562      WITH DEF_SCRATCH^ DO
1563
1564      BEGIN
1565
1566          PRIMARY          := RECORDS;
1567          SECONDARY        := BLOCK_SPAN;
1568          SWITCH           := BDATA[EDF$K_BLOCK_SPAN];
1569
1570          INSERT_IN_ORDER (REPLACE_OBJ);
1571
1572      END;      ( WITH DEF_SCRATCH^ DO )
1573
1574  END;      ( IF TRUE DESIGN_ORG = SEQUENTIAL )
1575
1576      { +
1577      CARRIAGE_CONTROL secondary.
1578      - )
1579      MAKE_SCRATCH;
1580
1581      WITH DEF_SCRATCH^ DO
1582
1583      BEGIN
1584
1585          PRIMARY          := RECORDS;
1586          SECONDARY        := CARRIAGE_CONTROL;
1587          QUALIFIER        := IDATA[EDF$K_CARR_CTRL];
1588
1589          INSERT_IN_ORDER (REPLACE_OBJ);
1590
1591  END;
1592
1593  IF (
1594      ((IDATA[EDF$K_SCRIPT_OPTION] = EDF$K_SEQ_DESIGN_FDL)
1595      OR
1596      (IDATA[EDF$K_SCRIPT_OPTION] = EDF$K_REL_DESIGN_FDL))
1597      AND
1598      (IDATA[EDF$K_RECORD_FORMAT] = FDL$C_VFC)
1599      ) THEN
1600
1601  BEGIN
```

```
1602      { +
1603      CONTROL_FIELD_SIZE secondary.
1604      - }
1605      MAKE_SCRATCH;
1606
1607      WITH DEF_SCRATCH^ DO
1608
1609      BEGIN
1610
1611          PRIMARY          := RECORDS;
1612          SECONDARY        := CONTROL_FIELD_SIZE;
1613          NUMBER           := IDATA[EDF$K_CONTROL_SIZE];
1614
1615          INSERT_IN_ORDER (REPLACE_OBJ);
1616
1617      END;
1618
1619      END;      { IF DESIGN_ORG = SEQ OR REL AND RECORD_FORMAT = VFC }
1620
1621      MAKE_SCRATCH;
1622
1623      WITH DEF_SCRATCH^ DO
1624
1625      BEGIN
1626
1627          { +
1628          FORMAT secondary.
1629          - }
1630          PRIMARY          := RECORDS;
1631          SECONDARY        := FORMAT;
1632          QUALIFIER        := IDATA[EDF$K_RECORD_FORMAT];
1633
1634          INSERT_IN_ORDER (REPLACE_OBJ);
1635
1636      END;      { WITH DEF_SCRATCH^ DO }
1637
1638      { +
1639      SIZE secondary.
1640      - }
1641      MAKE_SCRATCH;
1642
1643      WITH DEF_SCRATCH^ DO
1644
1645      BEGIN
1646
1647          PRIMARY          := RECORDS;
1648          SECONDARY        := SIZE;
1649          NUMBER           := IDATA[EDF$K_MAX_RECORD_SIZE];
1650
1651          INSERT_IN_ORDER (REPLACE_OBJ);
1652
1653      END;
1654
1655      END;      { NON_KEY_DEF }
1656
```

```
1658      ( ++
1659
1660      CALC_ALLOC -- Calculate the allocation for seq and rel files.
1661
1662      This routine handles the calculations for allocation for seq and rel files.
1663
1664      CALLING SEQUENCE:
1665
1666      ALLOC      := CALC_ALLOC (RECORD_TOT);
1667
1668      INPUT PARAMETERS:
1669
1670      RECORD_TOT
1671
1672      IMPLICIT INPUTS:
1673
1674      none
1675
1676      OUTPUT PARAMETERS:
1677
1678      none
1679
1680      IMPLICIT OUTPUTS:
1681
1682
1683      ROUTINES CALLED:
1684
1685      none
1686
1687      ROUTINE VALUE:
1688
1689      ALLOCATION CALCULATED
1690
1691      SIGNALS:
1692
1693      none
1694
1695      SIDE EFFECTS:
1696
1697      none
1698
1699      -- }
```



```
1701 FUNCTION CALC_ALLOC (RECORD_TOT : INTEGER) : INTEGER;  
1702  
1703 VAR  
1704     ALLOC      : INTEGER;  
1705     RATIO      : REAL;  
1706     BYTES_REAL : REAL;  
1707     NUMRECS_REAL : REAL;  
1708  
1709 BEGIN  
1710     ( +  
1711     Now let's figure out the allocation needed.  
1712     - )  
1713     BYTES_REAL      := RECORD_TOT;  
1714     NUMRECS_REAL    := IDATA[EDF$K_INITIAL_COUNT];  
1715  
1716     IF NUMRECS_REAL < 1.0 THEN  
1717         NUMRECS_REAL := 1.0;  
1718  
1719     RATIO := BYTES_REAL / 512.0;  
1720  
1721     IF (RATIO > (EDF$C_1GIGA / NUMRECS_REAL)) THEN  
1722  
1723         CALC_ALLOC := EDF$C_1GIGA  
1724  
1725     ELSE  
1726  
1727         CALC_ALLOC := ROUND (RATIO * NUMRECS_REAL);  
1728  
1729  
1730 END; ( CALC_ALLOC )  
1731
```

```
1733 { ++
1734
1735 SEQ_DEF -- Handle seq file stuff.
1736
1737 This routine handles the addition of the sequential file attributes.
1738
1739 CALLING SEQUENCE:
1740
1741 SEQ_DEF;
1742
1743 INPUT PARAMETERS:
1744
1745 none
1746
1747 IMPLICIT INPUTS:
1748
1749 none
1750
1751 OUTPUT PARAMETERS:
1752
1753 none
1754
1755 IMPLICIT OUTPUTS:
1756
1757 DEF_CURRENT
1758 DEF_HEAD
1759
1760 ROUTINES CALLED:
1761
1762 none
1763
1764 ROUTINE VALUE:
1765
1766 none
1767
1768 SIGNALS:
1769
1770 none
1771
1772 SIDE EFFECTS:
1773
1774 none
1775
1776 -- }
```

```
1778 PROCEDURE SEQ_DEF;
1779
1780 VAR
1781     ALLOC      : INTEGER;
1782     RECORD_TOT : INTEGER;
1783     RECORD_INT  : INTEGER;
1784     RECORD_REAL : REAL;
1785
1786 BEGIN
1787
1788     ( +
1789     Figure out how big each record is.
1790     - )
1791     RECORD_TOT      := 1DATA[EDF$K_MEAN_RECORD_SIZE];
1792
1793     IF VARIABLE_RECORDS THEN
1794         RECORD_TOT      := RECORD_TOT + 2;
1795
1796     ( +
1797     Assumes record size is less than 512 if BDATA[EDF$K_BLOCK_SPAN] is false.
1798     - )
1799     IF NOT BDATA[EDF$K_BLOCK_SPAN] THEN
1800         BEGIN
1801             ( +
1802             Increase the virtual size of each record so that it looks like
1803             an integer number of them fit in a block.
1804             - )
1805             RECORD_REAL := 512.0 / RECORD_TOT;
1806             RECORD_INT  := TRUNC (RECORD_REAL);
1807             RECORD_TOT  := 512 DIV RECORD_INT;
1808
1809         END;
1810
1811     END;
1812
1813     ALLOC      := CALC_ALLOC (RECORD_TOT);
1814
1815     ( +
1816     Now actually stuff the secondary from the above calculations.
1817     - )
1818     MAKE_SCRATCH;
1819
1820     WITH DEF_SCRATCH^ DO
1821         BEGIN
1822             ( +
1823             ALLOCATION secondary.
1824             - )
1825             PRIMARY      := FILES;
1826             SECONDARY    := ALLOCATION;
1827             NUMBER       := ALLOC;
1828
1829             INSERT_IN_ORDER (REPLACE_OBJ);
1830
1831         END;
1832     ( WITH DEF_SCRATCH^ DO )
1833
1834 END;
```

```
1835  
1836 MAKE_SCRATCH;  
1837  
1838 WITH DEF_SCRATCH^ DO  
1839  
1840 BEGIN  
1841  
1842     ( +  
1843     BEST_TRY_CONTIGUOUS secondary.  
1844     - )  
1845     PRIMARY          := FILES;  
1846     SECONDARY        := BEST_TRY_CONTIGUOUS;  
1847  
1848     INSERT_IN_ORDER (REPLACE_OBJ);  
1849  
1850 END;      ( WITH DEF_SCRATCH^ DO )  
1851  
1852 MAKE_SCRATCH;  
1853  
1854 WITH DEF_SCRATCH^ DO  
1855  
1856 BEGIN  
1857  
1858     ( +  
1859     EXTENSION secondary.  
1860     - )  
1861     PRIMARY          := FILES;  
1862     SECONDARY        := EXTENSION;  
1863     NUMBER           := ALLOC DIV 10;  
1864  
1865     INSERT_IN_ORDER (REPLACE_OBJ);  
1866  
1867 END;      ( WITH DEF_SCRATCH^ DO )  
1868  
1869 END;      ( SEQ_DEF )
```

```
1871 { **
1872
1873 REL_DEF -- Handle relative file stuff.
1874
1875 This routine handles the addition of the relative file attributes.
1876
1877 CALLING SEQUENCE:
1878
1879 REL_DEF;
1880
1881 INPUT PARAMETERS:
1882
1883 none
1884
1885 IMPLICIT INPUTS:
1886
1887 none
1888
1889 OUTPUT PARAMETERS:
1890
1891 none
1892
1893 IMPLICIT OUTPUTS:
1894
1895 DEF_CURRENT
1896 DEF_HEAD
1897
1898 ROUTINES CALLED:
1899
1900 none
1901
1902 ROUTINE VALUE:
1903
1904 none
1905
1906 SIGNALS:
1907
1908 none
1909
1910 SIDE EFFECTS:
1911
1912 none
1913
1914 -- }
```



```
1916 PROCEDURE REL_DEF;
1917
1918 VAR
1919     ALLOC           : INTEGER;
1920     RECORD_TOT      : INTEGER;
1921     BUCKET_TOT      : INTEGER;
1922     BUCKET          : INTEGER;
1923     RECS_PER_BUCKET : INTEGER;
1924     NUM_BUCKETS     : INTEGER;
1925
1926 BEGIN
1927
1928     { +
1929     See what the disk clustersize is.
1930     - }
1931     QUERY (EDFSK_CLUSTER_SIZE);
1932
1933     { +
1934     Calculate how large the bucketsize should be.
1935     Make them big enough for 16 records.
1936     - }
1937     RECORD_TOT      := IDATA[EDFSK_MAX_RECORD_SIZE] + 1;
1938
1939     IF VARIABLE_RECORDS THEN
1940
1941         RECORD_TOT      := RECORD_TOT + 2;
1942
1943     BUCKET_TOT      := 16 * RECORD_TOT;
1944
1945     BUCKET          := BUCKET_TOT DIV 512;
1946
1947     IF BUCKET < 1 THEN
1948
1949         BUCKET          := 1;
1950
1951     IF (BUCKET_TOT MOD 512) <> 0 THEN
1952
1953         BUCKET          := BUCKET + 1;
1954
1955     BUCKET          := MAX_FACTOR (IDATA[EDFSK_CLUSTER_SIZE],
1956                                   BUCKET, BKTSC_MAXBKTSIZ);
1957
1958     RECS_PER_BUCKET := (BUCKET * 512) DIV RECORD_TOT;
1959
1960     IF RECS_PER_BUCKET < 1 THEN
1961
1962         RECS_PER_BUCKET := 1;
1963
1964     NUM_BUCKETS     := IDATA[EDFSK_INITIAL_COUNT] DIV RECS_PER_BUCKET;
1965
1966     IF NUM_BUCKETS < 1 THEN
1967
1968         NUM_BUCKETS     := 1;
1969
1970     IF (IDATA[EDFSK_INITIAL_COUNT] MOD RECS_PER_BUCKET) <> 0 THEN
1971
1972         NUM_BUCKETS     := NUM_BUCKETS + 1;
```

```

1973 { +
1974 Add one more disk cluster into the allocation for the prolog.
1975 - )
1976 ALLOC := (BUCKET * NUM_BUCKETS) + IDATA[EDFSK_CLUSTER_SIZE];
1977
1978 { +
1979 Now actually stuff the secondary from the above calculations.
1980 - )
1981 MAKE_SCRATCH;
1982
1983 WITH DEF_SCRATCH^ DO
1984
1985 BEGIN
1986
1987     { +
1988     ALLOCATION secondary.
1989     - )
1990     PRIMARY := FILES;
1991     SECONDARY := ALLOCATION;
1992     NUMBER := ALLOC;
1993
1994     INSERT_IN_ORDER (REPLACE_OBJ);
1995
1996 END;      ( WITH DEF_SCRATCH^ DO )
1997
1998 MAKE_SCRATCH;
1999
2000 WITH DEF_SCRATCH^ DO
2001
2002 BEGIN
2003
2004     { +
2005     BEST_TRY_CONTIGUOUS secondary.
2006     - )
2007     PRIMARY := FILES;
2008     SECONDARY := BEST_TRY_CONTIGUOUS;
2009
2010     INSERT_IN_ORDER (REPLACE_OBJ);
2011
2012 END;      ( WITH DEF_SCRATCH^ DO )
2013
2014 MAKE_SCRATCH;
2015
2016 WITH DEF_SCRATCH^ DO
2017
2018 BEGIN
2019
2020     { +
2021     BUCKET_SIZE secondary.
2022     - )
2023     PRIMARY := FILES;
2024     SECONDARY := BUCKET_SIZE;
2025     NUMBER := BUCKET;
2026
2027     INSERT_IN_ORDER (REPLACE_OBJ);
2028

```

```
2030 END;      ( WITH DEF_SCRATCH^ DO )
2031
2032 MAKE_SCRATCH;
2033
2034 WITH DEF_SCRATCH^ DO
2035 BEGIN
2036     { +
2037     EXTENSION secondary.
2038     - }
2039     PRIMARY      := FILES;
2040     SECONDARY    := EXTENSION;
2041     NUMBER       := MAX_FACTOR (
2042                     BUCKET,
2043                     (ALLOC DIV 4),
2044                     EDF$C_1GIGA);
2045
2046     INSERT_IN_ORDER (REPLACE_OBJ);
2047
2048 END;      ( WITH DEF_SCRATCH^ DO )
2049
2050 MAKE_SCRATCH;
2051
2052 WITH DEF_SCRATCH^ DO
2053 BEGIN
2054     { +
2055     MAX_RECORD_NUMBER secondary.
2056     - }
2057     PRIMARY      := FILES;
2058     SECONDARY    := MAX_RECORD_NUMBER;
2059     NUMBER       := IDATA[EDF$R_INITIAL_COUNT];
2060
2061     INSERT_IN_ORDER (REPLACE_OBJ);
2062
2063 END;      ( WITH DEF_SCRATCH^ DO )
2064
2065 END;      ( REL_DEF )
2066
2067
2068
2069
```

```
2071 { ++
2072
2073 APPEND_DEF -- Add a key/areas def segment onto the end of the definition.
2074
2075 This routine puts all the attributes for a key and its areas onto the tail
2076 of the linked list.
2077
2078 CALLING SEQUENCE:
2079
2080 APPEND_DEF;
2081
2082 INPUT PARAMETERS:
2083
2084 none
2085
2086 IMPLICIT INPUTS:
2087
2088 none
2089
2090 OUTPUT PARAMETERS:
2091
2092 none
2093
2094 IMPLICIT OUTPUTS:
2095
2096 DEF_CURRENT
2097 DEF_HEAD
2098
2099 ROUTINES CALLED:
2100
2101 none
2102
2103 ROUTINE VALUE:
2104
2105 none
2106
2107 SIGNALS:
2108
2109 none
2110
2111 SIDE EFFECTS:
2112
2113 none
2114
2115 -- }
```

```
2117 PROCEDURE APPEND_DEF;
2118
2119 VAR
2120     DATA_AREA_NUMBER      : INTEGER;
2121     INDEX_AREA_NUMBER      : INTEGER;
2122     INIT_DATA_ALLOC        : INTEGER;
2123     INIT_INDEX_ALLOC        : INTEGER;
2124     ADDED_DATA_ALLOC        : INTEGER;
2125     ADDED_INDEX_ALLOC       : INTEGER;
2126     DATA_ALLOC             : INTEGER;
2127     INDEX_ALLOC             : INTEGER;
2128     DATA_EXT               : INTEGER;
2129     INDEX_EXT               : INTEGER;
2130     USED_DATA_BUCKETS       : INTEGER;
2131     UNUSED_DATA_BUCKETS     : INTEGER;
2132     USED_INDEX_BUCKETS      : INTEGER;
2133     UNUSED_INDEX_BUCKETS    : INTEGER;
2134     CHOSEN_DEPTH            : INTEGER;
2135     CHOSEN_DEPTH2           : INTEGER;
2136     TEMP_ALLOC              : INTEGER;
2137     I                        : INTEGER;
2138
2139 BEGIN
2140
2141     { +
2142     Get the user's decision on the value of the plotted file parameter.
2143     - }
2144     IF IDATA[EDF$K_SURFACE_OPTION] <> EDF$K_LINE_SURFACE THEN
2145
2146         CASE IDATA[EDF$K_SURFACE_OPTION] OF
2147
2148             EDF$K_FILL_SURFACE :      QUERY (EDF$K_DESIRED_FILL);
2149
2150             EDF$K_INIT_SURFACE :      QUERY (EDF$K_INITIAL_COUNT);
2151
2152             EDF$K_ADDED_SURFACE :      QUERY (EDF$K_ADDED_COUNT);
2153
2154             EDF$K_KEY_SURFACE :        ASK_KEY_SIZE;
2155
2156             EDF$K_SIZE_SURFACE :
2157
2158                 BEGIN
2159
2160                     ASK_MEAN_RECORD_SIZE;
2161
2162                     { +
2163                     Redo the SIZE secondary if this was a Record Size Surface.
2164                     - }
2165                     MAKE_SCRATCH;
2166
2167                     WITH DEF_SCRATCH^ DO
2168
2169                         BEGIN
2170
2171                             PRIMARY                := RECORDS;
2172                             SECONDARY              := SIZE;
2173                             NUMBER                  := IDATA[EDF$K_MAX_RECORD_SIZE];
```



```
2174
2175         INSERT_IN_ORDER (REPLACE_OBJ);
2176
2177         END;      { WITH DEF_SCRATCH^ DO }
2178
2179     END;          { SIZE_SURFACE }
2180
2181 OTHERWISE
2182
2183     { NULL-STATEMENT } ;
2184
2185     END;      { CASE }
2186
2187 { +
2188 See what bucket size the user chose and recalculate the depth
2189 based on that bucket size alone. Find out the most reasonable
2190 bucket size default by looking for the left end of the 'natural depth'.
2191 The primary_buckets arrays are reset to zero now as well.
2192 - }
2193 BUCKET_DEFAULT      := NATURAL_DEPTH;
2194
2195 QUERY (EDFSK_BLOCKS_IN_BUCKET);
2196
2197 FOR I := 0 TO 31 DO
2198
2199 BEGIN
2200
2201     INIT_PRIMARY_BUCKETS[I]      := 0;
2202     ADDED_PRIMARY_BUCKETS[I]     := 0;
2203
2204 END;
2205
2206 CHOSEN_DEPTH      := PROLOGUE3_DEPTH;
2207
2208 { +
2209 Now finish getting the info to flesh out the FDL definition.
2210 - }
2211 QUERY (EDFSK_KEY_CHANGES);
2212 QUERY (EDFSK_KEY_NAME);
2213
2214 { +
2215 Figure the index allocation at the same time, though.
2216 - }
2217 INIT_DATA_ALLOC      := INIT_NUMBER_BUCKETS[0];
2218 ADDED_DATA_ALLOC     := ADDED_NUMBER_BUCKETS[0];
2219
2220 { +
2221 Find total number of buckets in index.
2222 - }
2223 INIT_INDEX_ALLOC     := 0;
2224 ADDED_INDEX_ALLOC    := 0;
2225
2226 FOR I := 1 TO CHOSEN_DEPTH DO
2227
2228 BEGIN
2229
2230     INIT_INDEX_ALLOC      := INIT_INDEX_ALLOC + INIT_NUMBER_BUCKETS[I];
```

```
2231      ADDED_INDEX_ALLOC      := ADDED_INDEX_ALLOC + ADDED_NUMBER_BUCKETS[I];
2232
2233  END;
2234
2235  { +
2236  Now merge any additional records into the existing ones.
2237  - }
2238  IF IDATA[EDFSK_ADDED_COUNT] <> 0 THEN
2239
2240  BEGIN
2241
2242      USED_DATA_BUCKETS      :=
2243          TRUNC (RDATA[EDFSK_LOAD_FILL] * INIT_DATA_ALLOC) + 1;
2244      USED_INDEX_BUCKETS      :=
2245          TRUNC (RDATA[EDFSK_LOAD_FILL] * INIT_INDEX_ALLOC) + 1;
2246      UNUSED_DATA_BUCKETS      := INIT_DATA_ALLOC - USED_DATA_BUCKETS;
2247      UNUSED_INDEX_BUCKETS      := INIT_INDEX_ALLOC - USED_INDEX_BUCKETS;
2248
2249      IF ADDED_DATA_ALLOC > UNUSED_DATA_BUCKETS THEN
2250
2251          ADDED_DATA_ALLOC      := ADDED_DATA_ALLOC - UNUSED_DATA_BUCKETS
2252
2253      ELSE
2254
2255          ADDED_DATA_ALLOC      := 0;
2256
2257      IF ADDED_INDEX_ALLOC > UNUSED_INDEX_BUCKETS THEN
2258
2259          ADDED_INDEX_ALLOC      := ADDED_INDEX_ALLOC - UNUSED_INDEX_BUCKETS
2260
2261      ELSE
2262
2263          ADDED_INDEX_ALLOC      := 0;
2264
2265      IF ADDED_DATA_ALLOC > 0 THEN
2266
2267          INIT_DATA_ALLOC      := INIT_DATA_ALLOC + ADDED_DATA_ALLOC;
2268
2269      IF ADDED_INDEX_ALLOC > 0 THEN
2270
2271          INIT_INDEX_ALLOC      := INIT_INDEX_ALLOC + ADDED_INDEX_ALLOC;
2272
2273  END;      { IF TRUE IDATA[EDFSK_ADDED_COUNT] <> 0 }
2274
2275  { +
2276  Calc to get total number of blocks for that many buckets.
2277  And also round the allocations 'slightly' up.
2278  Double check boundaries to prevent integer overflows. Enforce max of 1Giga.
2279  - }
2280  IF INIT_DATA_ALLOC > (EDFSC_1GIGA DIV IDATA[EDFSK_BLOCKS_IN_BUCKET]) THEN
2281
2282      DATA_ALLOC      := EDFSC_1GIGA
2283
2284  ELSE
2285
2286      DATA_ALLOC      := INIT_DATA_ALLOC * IDATA[EDFSK_BLOCKS_IN_BUCKET];
2287
```

```
2288 IF INIT_INDEX_ALLOC >
2289     (EDF$C_1GIGA DIV IDATA[EDF$K_BLOCKS_IN_BUCKET]) THEN
2290
2291     INDEX_ALLOC      := EDF$C_1GIGA
2292
2293 ELSE
2294
2295     INDEX_ALLOC      := INIT_INDEX_ALLOC + IDATA[EDF$K_BLOCKS_IN_BUCKET];
2296
2297 { +
2298 Since we're just about to allocate the user's file based on multiple
2299 areas, get rid of any existing secondaries that would be confusing.
2300 - }
2301 POINT_AT_DEFINITION;
2302
2303 IF FIND_OBJECT (PRI,FILES,0,DUMMY_SECONDARY$,0) THEN
2304
2305 BEGIN
2306
2307     REPEAT
2308
2309         IF (
2310             (DEF_CURRENT^.PRIMARY = FILES)
2311             AND
2312             (DEF_CURRENT^.SECONDARY IN [ ALLOCATION, EXTENSION,
2313             BUCKET_SIZE, BEST_TRY_CONTIGUOUS, CLUSTER_SIZE ])
2314         ) THEN
2315
2316             DELETE_CURRENT
2317
2318         ELSE
2319
2320             INCR_CURRENT;
2321
2322     UNTIL (DEF_CURRENT = NIL) OR (DEF_CURRENT^.PRIMARY <> FILES);
2323
2324 END;      { IF TRUE FIND_OBJECT (FILES) }
2325
2326 { +
2327 Compute the correct area numbers.
2328 - }
2329 IF IDATA[EDF$K_ACTIVE_KEY] < 127 THEN
2330
2331     DATA_AREA_NUMBER      := (2*IDATA[EDF$K_ACTIVE_KEY])
2332
2333 ELSE
2334
2335     DATA_AREA_NUMBER      := 254;
2336
2337 INDEX_AREA_NUMBER          := DATA_AREA_NUMBER + 1;
2338
2339 { +
2340 Make the area primary.
2341 - }
2342 MAKE_SCRATCH;
2343
2344 WITH DEF_SCRATCH^ DO
```

```
2345 BEGIN
2346
2347 { +
2348 AREA = primary (for data).
2349 - }
2350 OBJECT TYPE      := PRI;
2351 PRIMARY          := AREA;
2352 PRINUM           := DATA_AREA_NUMBER;
2353
2354 INSERT_IN_ORDER (REPLACE_OBJ);
2355
2356 END;      ( WITH DEF_SCRATCH^ DO )
2357
2358 { +
2359 Now actually stuff the secondary from the above calculations.
2360 - }
2361 IF IDATA[EDFSK_ACTIVE_KEY] < 127 THEN
2362     TEMP_ALLOC      := 0
2363
2364 ELSE IF FIND_OBJECT (SEC,AREA,254,ALLOCATIONS,0) THEN
2365     TEMP_ALLOC      := DEF_CURRENT^.NUMBER
2366
2367 ELSE
2368     TEMP_ALLOC      := 0;
2369
2370 MAKE_SCRATCH;
2371
2372 WITH DEF_SCRATCH^ DO
2373 BEGIN
2374     { +
2375     ALLOCATION secondary (for data area).
2376     - }
2377     PRIMARY          := AREA;
2378     PRINUM           := DATA_AREA_NUMBER;
2379     SECONDARY        := ALLOCATIONS;
2380
2381     NUMBER           := DATA_ALLOC + TEMP_ALLOC;
2382
2383     INSERT_IN_ORDER (REPLACE_OBJ);
2384
2385 END;      ( WITH DEF_SCRATCH^ DO )
2386
2387 MAKE_SCRATCH;
2388
2389 WITH DEF_SCRATCH^ DO
2390 BEGIN
2391     { +
2392     BEST_TRY_CONTIGUOUS secondary (for data area).
2393     - }
2400
2401
```

```
2402     PRIMARY          := AREA;
2403     PRINUM            := DATA_AREA_NUMBER;
2404     SECONDARY         := BEST_TRY_CONTIGUOUS;
2405
2406     INSERT_IN_ORDER (REPLACE_OBJ);
2407
2408     END;             ( WITH DEF_SCRATCH^ DO )
2409
2410     MAKE_SCRATCH;
2411
2412     WITH DEF_SCRATCH^ DO
2413
2414     BEGIN
2415
2416         { +
2417         BUCKET_SIZE secondary (for data area).
2418         - }
2419         PRIMARY          := AREA;
2420         PRINUM            := DATA_AREA_NUMBER;
2421         SECONDARY         := BUCKET_SIZES;
2422         NUMBER            := IDATA[EDF$K_BLOCKS_IN_BUCKET];
2423
2424         INSERT_IN_ORDER (REPLACE_OBJ);
2425
2426         END;             ( WITH DEF_SCRATCH^ DO )
2427
2428         MAKE_SCRATCH;
2429
2430         WITH DEF_SCRATCH^ DO
2431
2432         BEGIN
2433
2434             { +
2435             EXTENSION secondary (for data area).
2436             - }
2437             PRIMARY          := AREA;
2438             PRINUM            := DATA_AREA_NUMBER;
2439             SECONDARY         := EXTENSIONS;
2440             NUMBER            := MAX_FACTOR (
2441                 IDATA[EDF$K_BLOCKS_IN_BUCKET],
2442                 ((DATA_ALLOC+TEMP_ALLOC) DIV 4),
2443                 EDF$C_TGIGA);
2444
2445             INSERT_IN_ORDER (REPLACE_OBJ);
2446
2447             END;             ( WITH DEF_SCRATCH^ DO )
2448
2449             MAKE_SCRATCH;
2450
2451             WITH DEF_SCRATCH^ DO
2452
2453             BEGIN
2454
2455                 { +
2456                 AREA n primary (for index).
2457                 - }
2458                 OBJECT_TYPE      := PRI;
```



```
2459     PRIMARY          := AREA;
2460     PRINUM            := INDEX_AREA_NUMBER;
2461
2462     INSERT_IN_ORDER (REPLACE_OBJ);
2463
2464     END;          ( WITH DEF_SCRATCH^ DO )
2465
2466     MAKE_SCRATCH;
2467
2468     IF IDATA[EDF$K_ACTIVE_KEY] < 127 THEN
2469         TEMP_ALLOC      := 0
2470     ELSE IF FIND_OBJECT (SEC,AREA,255,ALLOCATIONS,0) THEN
2471         TEMP_ALLOC      := DEF_CURRENT^.NUMBER
2472     ELSE
2473         TEMP_ALLOC      := 0;
2474
2475     WITH DEF_SCRATCH^ DO
2476     BEGIN
2477         { +
2478         ALLOCATION secondary (for index area).
2479         - }
2480         PRIMARY          := AREA;
2481         PRINUM            := INDEX_AREA_NUMBER;
2482         SECONDARY         := ALLOCATIONS;
2483         NUMBER            := INDEX_ALLOC + TEMP_ALLOC;
2484
2485         INSERT_IN_ORDER (REPLACE_OBJ);
2486
2487     END;          ( WITH DEF_SCRATCH^ DO )
2488
2489     MAKE_SCRATCH;
2490
2491     WITH DEF_SCRATCH^ DO
2492     BEGIN
2493         { +
2494         BEST_TRY_CONTIGUOUS secondary (for index area).
2495         - }
2496         PRIMARY          := AREA;
2497         PRINUM            := INDEX_AREA_NUMBER;
2498         SECONDARY         := BEST_TRY_CONTIGUOUS;
2499
2500         INSERT_IN_ORDER (REPLACE_OBJ);
2501
2502     END;          ( WITH DEF_SCRATCH^ DO )
2503
2504     MAKE_SCRATCH;
2505
2506     WITH DEF_SCRATCH^ DO
```

```
2516 BEGIN
2517
2518 { +
2519 BUCKET_SIZE secondary (for index area).
2520 - }
2521 PRIMARY := AREA;
2522 PRINUM := INDEX_AREA_NUMBER;
2523 SECONDARY := BUCKET_SIZE;
2524 NUMBER := IDATA[EDFSK_BLOCKS_IN_BUCKET];
2525
2526 INSERT_IN_ORDER (REPLACE_OBJ);
2527
2528 END; { WITH DEF_SCRATCH^ DO }
2529
2530 MAKE_SCRATCH;
2531
2532 WITH DEF_SCRATCH^ DO
2533 BEGIN
2534
2535 { +
2536 EXTENSION secondary (for index area).
2537 - }
2538 PRIMARY := AREA;
2539 PRINUM := INDEX_AREA_NUMBER;
2540 SECONDARY := EXTENSIONS;
2541 NUMBER := MAX_FACTOR (
2542 IDATA[EDFSK_BLOCKS_IN_BUCKET],
2543 ((INDEX_ALLOC+TEMP_ALLOC) DIV 4),
2544 EDFSC_1GIGA);
2545
2546 INSERT_IN_ORDER (REPLACE_OBJ);
2547
2548 END; { WITH DEF_SCRATCH^ DO }
2549
2550 MAKE_SCRATCH;
2551
2552 WITH DEF_SCRATCH^ DO
2553 BEGIN
2554
2555 { +
2556 KEY n primary.
2557 - }
2558 OBJECT_TYPE := PRI;
2559 PRINUM := IDATA[EDFSK_ACTIVE_KEY];
2560
2561 INSERT_IN_ORDER (REPLACE_OBJ);
2562
2563 END; { WITH DEF_SCRATCH^ DO }
2564
2565 MAKE_SCRATCH;
2566
2567 WITH DEF_SCRATCH^ DO
2568 BEGIN
2569
2570
2571
2572
```

```
2573
2574      { +
2575      CHANGES secondary.
2576      - }
2577      PRINUM          := IDATA[EDF$K_ACTIVE_KEY];
2578      SECONDARY       := CHANGES;
2579      SWITCH          := BDATA[EDF$K_KEY_CHANGES];
2580
2581      INSERT_IN_ORDER (REPLACE_OBJ);
2582
2583  END;      ( WITH DEF_SCRATCH^ DO )
2584
2585  MAKE_SCRATCH;
2586
2587  WITH DEF_SCRATCH^ DO
2588
2589  BEGIN
2590
2591      { +
2592      DATA_AREA secondary.
2593      - }
2594      PRINUM          := IDATA[EDF$K_ACTIVE_KEY];
2595      SECONDARY       := DATA_AREA;
2596      NUMBER          := DATA_AREA_NUMBER;
2597
2598      INSERT_IN_ORDER (REPLACE_OBJ);
2599
2600  END;      ( WITH DEF_SCRATCH^ DO )
2601
2602  MAKE_SCRATCH;
2603
2604  WITH DEF_SCRATCH^ DO
2605
2606  BEGIN
2607
2608      { +
2609      DATA_FILL secondary.
2610      - }
2611      PRINUM          := IDATA[EDF$K_ACTIVE_KEY];
2612      SECONDARY       := DATA_FILL;
2613      NUMBER          := IDATA[EDF$K_FDL_FILL];
2614
2615      INSERT_IN_ORDER (REPLACE_OBJ);
2616
2617  END;      ( WITH DEF_SCRATCH^ DO )
2618
2619  MAKE_SCRATCH;
2620
2621  WITH DEF_SCRATCH^ DO
2622
2623  BEGIN
2624
2625      { +
2626      DATA_KEY_COMPRESSION secondary.
2627      - }
2628      PRINUM          := IDATA[EDF$K_ACTIVE_KEY];
2629      SECONDARY       := DATA_KEY_COMPRESSION;
```

```
2630      SWITCH                := BDATA[EDFSK_KEY_COMP_WANTED];
2631
2632      INSERT_IN_ORDER (REPLACE_OBJ);
2633
2634      END;      { WITH DEF_SCRATCH^ DO }
2635
2636      MAKE_SCRATCH;
2637
2638      IF IDATA[EDFSK_ACTIVE_KEY] = 0 THEN
2639
2640      BEGIN
2641
2642          WITH DEF_SCRATCH^ DO
2643
2644          BEGIN
2645
2646              { +
2647              DATA_RECORD_COMPRESSION secondary.
2648              - }
2649              PRINUM                := IDATA[EDFSK_ACTIVE_KEY];
2650              SECONDARY              := DATA_RECORD_COMPRESSION;
2651              SWITCH                := BDATA[EDFSK_REC_COMP_WANTED];
2652
2653              INSERT_IN_ORDER (REPLACE_OBJ);
2654
2655          END;      { WITH DEF_SCRATCH^ DO }
2656
2657          MAKE_SCRATCH;
2658
2659      END;      { IDATA[EDFSK_ACTIVE_KEY] = 0 }
2660
2661      WITH DEF_SCRATCH^ DO
2662
2663      BEGIN
2664
2665          { +
2666          DUPLICATES secondary.
2667          - }
2668          PRINUM                := IDATA[EDFSK_ACTIVE_KEY];
2669          SECONDARY              := DUPLICATES;
2670          SWITCH                := BDATA[EDFSK_KEY_DUPS];
2671
2672          INSERT_IN_ORDER (REPLACE_OBJ);
2673
2674      END;      { WITH DEF_SCRATCH^ DO }
2675
2676      MAKE_SCRATCH;
2677
2678      WITH DEF_SCRATCH^ DO
2679
2680      BEGIN
2681
2682          { +
2683          INDEX_AREA secondary.
2684          - }
2685          PRINUM                := IDATA[EDFSK_ACTIVE_KEY];
2686          SECONDARY              := INDEX_AREA;
```

```
2687      NUMBER                := INDEX_AREA_NUMBER;
2688
2689      INSERT_IN_ORDER (REPLACE_OBJ);
2690
2691      END;      { WITH DEF_SCRATCH^ DO }
2692
2693      MAKE_SCRATCH;
2694
2695      WITH DEF_SCRATCH^ DO
2696
2697      BEGIN
2698
2699      { +
2700      INDEX_FILL secondary.
2701      - }
2702      PRINUM                := IDATA[EDFSK_ACTIVE_KEY];
2703      SECONDARY              := INDEX_FILL;
2704      NUMBER                := IDATA[EDFSK_FDL_FILL];
2705
2706      INSERT_IN_ORDER (REPLACE_OBJ);
2707
2708      END;      { WITH DEF_SCRATCH^ DO }
2709
2710      MAKE_SCRATCH;
2711
2712      WITH DEF_SCRATCH^ DO
2713
2714      BEGIN
2715
2716      { +
2717      INDEX_COMPRESSION secondary.
2718      - }
2719      PRINUM                := IDATA[EDFSK_ACTIVE_KEY];
2720      SECONDARY              := INDEX_COMPRESSION;
2721      SWITCH                := BDATA[EDFSK_IDX_COMP_WANTED];
2722
2723      INSERT_IN_ORDER (REPLACE_OBJ);
2724
2725      END;      { WITH DEF_SCRATCH^ DO }
2726
2727      IF NOT BDATA[EDFSK_SEGMENTED] THEN
2728
2729      BEGIN
2730
2731      MAKE_SCRATCH;
2732
2733      WITH DEF_SCRATCH^ DO
2734
2735      BEGIN
2736
2737      { +
2738      LENGTH secondary.
2739      - }
2740      PRINUM                := IDATA[EDFSK_ACTIVE_KEY];
2741      SECONDARY              := SEG_LENGTH;
2742      NUMBER                := IDATA[EDFSK_KEY_SIZE];
2743
```



```
2744      INSERT_IN_ORDER (REPLACE_OBJ);
2745
2746      END;
2747
2748      END      { IF TRUE NOT SEGMENTED }
2749
2750      ELSE
2751
2752      FOR SEGMENT_NUMBER := 0 TO 7 DO
2753
2754      BEGIN
2755
2756          IF SEGMENT_WANTED[SEGMENT_NUMBER] THEN
2757
2758              BEGIN
2759
2760                  MAKE_SCRATCH;
2761
2762                  WITH DEF_SCRATCH^ DO
2763
2764                      BEGIN
2765
2766                          { +
2767                          LENGTH secondary.
2768                          - }
2769                          PRINUM          := IDATA[EDF$K_ACTIVE_KEY];
2770                          SECONDARY       := SEG_LENGTH;
2771                          NUMBER          := SEGMENT_LENGTH[SEGMENT_NUMBER];
2772                          SECNUM          := SEGMENT_NUMBER;
2773
2774                          INSERT_IN_ORDER (REPLACE_OBJ);
2775
2776                      END;
2777
2778                  END;
2779
2780              END;      { IF TRUE BDATA[EDF$K_SEGMENTED] }
2781
2782              MAKE_SCRATCH;
2783
2784              WITH DEF_SCRATCH^ DO
2785
2786                  BEGIN
2787
2788                      { +
2789                      LEVEL1_INDEX_AREA secondary.
2790                      - }
2791                      PRINUM          := IDATA[EDF$K_ACTIVE_KEY];
2792                      SECONDARY       := LEVEL1_INDEX_AREA;
2793                      NUMBER          := INDEX_AREA_NUMBER;
2794
2795                      INSERT_IN_ORDER (REPLACE_OBJ);
2796
2797                  END;      { WITH DEF_SCRATCH^ DO }
2798
2799              { +
2800              NAME secondary.
```

```
2801 - }
2802 IF BDATA[EDFSK_KEY_NAME] THEN
2803
2804 BEGIN
2805     MAKE_SCRATCH;
2806     WITH DEF_SCRATCH^ DO
2807     BEGIN
2808         LIB$COPY_DXDX (SDATA[EDFSK_KEY_NAME],STRING);
2809         STR$FREE1_DX (SDATA[EDFSK_KEY_NAME]);
2810
2811         PRINUM                := IDATA[EDFSK_ACTIVE_KEY];
2812         SECONDARY              := NAMES;
2813
2814         INSERT_IN_ORDER (REPLACE_OBJ);
2815     END; { WITH DEF_SCRATCH^ }
2816 END { IF TRUE BDATA[EDFSK_KEY_NAME] }
2817
2818 ELSE
2819 BEGIN
2820     IF FIND_OBJECT (SEC,KEY,IDATA[EDFSK_ACTIVE_KEY],NAMES,0) THEN
2821         DELETE_CURRENT;
2822     END; { IF FALSE BDATA[EDFSK_KEY_NAME] }
2823
2824 IF (
2825     (IDATA[EDFSK_ACTIVE_KEY] = 0)
2826     AND
2827     (VDATA[EDFSK_PROLOGUE_VERSION])
2828 ) THEN
2829 BEGIN
2830     MAKE_SCRATCH;
2831     WITH DEF_SCRATCH^ DO
2832     BEGIN
2833         { +
2834         PROLOGUE secondary.
2835         - }
2836         PRINUM                := IDATA[EDFSK_ACTIVE_KEY]; { = 0 }
2837         SECONDARY              := PROLOGUE;
2838         NUMBER                 := IDATA[EDFSK_PROLOGUE_VERSION];
2839
2840         INSERT_IN_ORDER (REPLACE_OBJ);
2841     END; { WITH DEF_SCRATCH^ DO }
2842
2843
```

```
2858 END; ( IF (IDATA[EDF$K_ACTIVE_KEY] = 0) AND (VDATA[EDF$K_PROLOGUE_VERSION]) )
2859
2860 IF NOT BDATA[EDF$K_SEGMENTED] THEN
2861
2862 BEGIN
2863
2864     MAKE_SCRATCH;
2865
2866     WITH DEF_SCRATCH^ DO
2867
2868     BEGIN
2869
2870         ( +
2871         POSITION secondary.
2872         - )
2873         PRINUM          := IDATA[EDF$K_ACTIVE_KEY];
2874         SECONDARY        := SEG_POSITION;
2875         NUMBER           := IDATA[EDF$K_KEY_POSITION];
2876
2877         INSERT_IN_ORDER (REPLACE_OBJ);
2878
2879     END;
2880
2881 END      ( IF TRUE NOT SEGMENTED )
2882
2883 ELSE
2884
2885 FOR SEGMENT_NUMBER := 0 TO 7 DO
2886
2887 BEGIN
2888
2889     IF SEGMENT_WANTED[SEGMENT_NUMBER] THEN
2890
2891     BEGIN
2892
2893         MAKE_SCRATCH;
2894
2895         WITH DEF_SCRATCH^ DO
2896
2897         BEGIN
2898
2899             ( +
2900             POSITION secondary.
2901             - )
2902             PRINUM          := IDATA[EDF$K_ACTIVE_KEY];
2903             SECONDARY        := SEG_POSITION;
2904             NUMBER           := SEGMENT_POSITION[SEGMENT_NUMBER];
2905             SECNUM           := SEGMENT_NUMBER;
2906
2907             INSERT_IN_ORDER (REPLACE_OBJ);
2908
2909         END;
2910
2911     END;
2912
2913 END;      ( IF TRUE BDATA[EDF$K_SEGMENTED] )
2914
```

```
2915
2916 { +
2917 TYPE secondary.
2918 - }
2919 MAKE_SCRATCH;
2920
2921 WITH DEF_SCRATCH^ DO
2922
2923 BEGIN
2924
2925     PRINUM           := IDATA[EDFSK_ACTIVE_KEY];
2926     SECONDARY        := SEG_TYPE;
2927     QUALIFIER        := IDATA[EDFSK_KEY_TYPE];
2928
2929     { +
2930     Make type the last secondary in the key primary.
2931     - }
2932     SECNUM           := 7;
2933
2934     INSERT_IN_ORDER (REPLACE_OBJ);
2935
2936 END;      { WITH DEF_SCRATCH^ DO }
2937
2938 { +
2939 After the user has chosen his bucket size, ask about
2940 global buffers.
2941 - }
2942 IF IDATA[EDFSK_ACTIVE_KEY] = 0 THEN
2943
2944 BEGIN
2945
2946     ASK_GLOBAL_WANTED;
2947
2948     { +
2949     GLOBAL_BUFFER_COUNT secondary.
2950     - }
2951     IF BDATA[EDFSK_GLOBAL_WANTED] THEN
2952
2953     BEGIN
2954
2955         MAKE_SCRATCH;
2956
2957         WITH DEF_SCRATCH^ DO
2958
2959         BEGIN
2960
2961             PRIMARY      := FILES;
2962             SECONDARY     := GLOBAL_BUFFER_COUNT;
2963             NUMBER        := IDATA[EDFSK_GLOBAL_COUNT];
2964
2965             INSERT_IN_ORDER (REPLACE_OBJ);
2966
2967         END;      { WITH DEF_SCRATCH^ DO }
2968
2969     END          { IF TRUE BDATA[EDFSK_GLOBAL_WANTED] }
2970
2971 ELSE
```

```
2972
2973 BEGIN
2974
2975     IF FIND_OBJECT (SEC,FILES$,0,GLOBAL_BUFFER_COUNT,0) THEN
2976         DELETE_CURRENT;
2977
2978     END;    { IF FALSE BDATA[EDFSK_GLOBAL_WANTED] }
2979
2980 END;    { IF TRUE IDATA[EDFSK_ACTIVE_KEY] = 0 }
2981
2982 { +
2983 Show the user what he has.
2984 - }
2985 CHOSEN_DEPTH2      := CHOSEN_DEPTH + 1;
2986
2987 IF NOT AUTO_TUNE THEN
2988 BEGIN
2989
2990     WRITELN (
2991         CRLF,
2992         SHIFT, 'The Depth of Key', IDATA[EDFSK_ACTIVE_KEY]:3,
2993         ' is Estimated to be No Greater', CRLF_SHIFT,
2994         ' than '
2995         CHOSEN_DEPTH:NUM_LEN(CHOSEN_DEPTH), ' Index levels, which is ',
2996         CHOSEN_DEPTH2:NUM_LEN(CHOSEN_DEPTH2), ' Total levels.'
2997     );
2998
2999     QUERY (EDFSK_RETURN);
3000
3001     QUERY (EDFSK_RETURN);
3002
3003 END;
3004
3005 END;    { APPEND_DEF }
```



```
3007      ( ++
3008
3009      LINK_RESULTS -- Incorporate the 'designed' variables into the Linked List.
3010
3011      This routine puts the final state of the variables into the Definition.
3012
3013      CALLING SEQUENCE:
3014
3015      LINK_RESULTS:
3016
3017      INPUT PARAMETERS:
3018
3019      none
3020
3021      IMPLICIT INPUTS:
3022
3023      none
3024
3025      OUTPUT PARAMETERS:
3026
3027      none
3028
3029      IMPLICIT OUTPUTS:
3030
3031      DEF_CURRENT
3032      DEF_HEAD
3033
3034      ROUTINES CALLED:
3035
3036      none
3037
3038      ROUTINE VALUE:
3039
3040      none
3041
3042      SIGNALS:
3043
3044      none
3045
3046      SIDE EFFECTS:
3047
3048      none
3049
3050      -- )
```

```
3052  PROCEDURE LINK_RESULTS;  
3053  
3054  BEGIN  
3055  
3056      { +  
3057      Put the terminal back.  
3058      - }  
3059      EDF$RESET_SCROLL;  
3060      CLEAR (SCREEN);  
3061      VISIBLE_QUESTION := FALSE;  
3062      WAIT_HELP := FALSE;  
3063      TAKE_DEFAULTS := TRUE;  
3064  
3065      { +  
3066      If this is the 1st time through, get the general file attributes.  
3067      - }  
3068      IF [DATA[EDF$K_ACTIVE_KEY] = 0 THEN  
3069          NON_KEY_DEF;  
3070  
3071      { +  
3072      Add this key's data to the linked list.  
3073      - }  
3074      APPEND_DEF;  
3075  
3076      LINKED := TRUE;  
3077  
3078  END; { LINK_RESULTS }  
3079
```

```
3081      { **
3082
3083      MERGE_AREA -- Collapse area definitions onto one another.
3084
3085      This routine updates the area sections after adding together allocations.
3086
3087      CALLING SEQUENCE:
3088
3089      MERGE_AREA (CURKEY,MAXKEY,SRCDATA,DSTDATA,SRCIDX,DSTIDX);
3090
3091      INPUT PARAMETERS:
3092
3093      CURKEY
3094      MAXKEY
3095      SRCDATA
3096      DSTDATA
3097      SRCIDX
3098      DSTIDX
3099
3100      IMPLICIT INPUTS:
3101
3102      DEF_CURRENT
3103      DEF_HEAD
3104
3105      OUTPUT PARAMETERS:
3106
3107      none
3108
3109      IMPLICIT OUTPUTS:
3110
3111      DEF_CURRENT
3112      DEF_HEAD
3113
3114      ROUTINES CALLED:
3115
3116      none
3117
3118      ROUTINE VALUE:
3119
3120      none
3121
3122      SIGNALS:
3123
3124      none
3125
3126      SIDE EFFECTS:
3127
3128      none
3129
3130      -- }
```

```
3132 PROCEDURE MERGE_AREA (CURKEY,MAXKEY,SRCDATA,DSTDATA,SRCIDX,DSTIDX : INTEGER);
```

```
3133 VAR
```

```
3134     KEYNUM           : INTEGER;  
3135     SOURCE_DATA_BUCKET : INTEGER;  
3136     SOURCE_DATA_ALLOC : INTEGER;  
3137     SOURCE_DATA_EXT   : INTEGER;  
3138     SOURCE_INDEX_BUCKET : INTEGER;  
3139     SOURCE_INDEX_ALLOC : INTEGER;  
3140     SOURCE_INDEX_EXT   : INTEGER;  
3141
```

```
3142 BEGIN
```

```
3143     { +  
3144     Set up the defaults in case some line_objects are not found.  
3145     - }
```

```
3146     SOURCE_DATA_BUCKET      := 3;  
3147     SOURCE_DATA_ALLOC      := 0;  
3148     SOURCE_DATA_EXT        := 0;  
3149     SOURCE_INDEX_BUCKET    := 3;  
3150     SOURCE_INDEX_ALLOC     := 0;  
3151     SOURCE_INDEX_EXT       := 0;  
3152
```

```
3153     { +  
3154     Get the bucket sizes, allocations, and extensions of the areas that  
3155     are going away.
```

```
3156     THESE COULD ALL BE OPTIMIZED BY REALIZING THAT THEY'RE ALL  
3157     ADJACENT LINE_OBJECTS!!!
```

```
3158     - }  
3159     IF FIND_OBJECT (SEC,AREA,SRCDATA,BUCKET_SIZES,0) THEN
```

```
3160         SOURCE_DATA_BUCKET      := DEF_CURRENT^.NUMBER;
```

```
3161     IF FIND_OBJECT (SEC,AREA,SRCDATA,ALLOCATIONS,0) THEN
```

```
3162         SOURCE_DATA_ALLOC      := DEF_CURRENT^.NUMBER;
```

```
3163     IF FIND_OBJECT (SEC,AREA,SRCDATA,EXTENSIONS,0) THEN
```

```
3164         SOURCE_DATA_EXT        := DEF_CURRENT^.NUMBER;
```

```
3165     IF FIND_OBJECT (SEC,AREA,SRCIDX,BUCKET_SIZES,0) THEN
```

```
3166         SOURCE_INDEX_BUCKET    := DEF_CURRENT^.NUMBER;
```

```
3167     IF FIND_OBJECT (SEC,AREA,SRCIDX,ALLOCATIONS,0) THEN
```

```
3168         SOURCE_INDEX_ALLOC     := DEF_CURRENT^.NUMBER;
```

```
3169     IF FIND_OBJECT (SEC,AREA,SRCIDX,EXTENSIONS,0) THEN
```

```
3170         SOURCE_INDEX_EXT       := DEF_CURRENT^.NUMBER;
```

```
3171     { +  
3172
```

```
3189   If SRC = DST, then ignore the above numbers.
3190   - )
3191   IF SRCDATA = DSTDATA THEN
3192   BEGIN
3193       SOURCE_DATA_BUCKET           := 0;
3194       SOURCE_DATA_ALLOC           := 0;
3195       SOURCE_DATA_EXT             := 0;
3196   END;
3197   IF SRCIDX = DSTIDX THEN
3198   BEGIN
3199       SOURCE_INDEX_BUCKET           := 0;
3200       SOURCE_INDEX_ALLOC           := 0;
3201       SOURCE_INDEX_EXT             := 0;
3202   END;
3203   { +
3204   Now add these to the areas that we're merging into.
3205   Bucket sizes get maximized.
3206   - )
3207   IF FIND_OBJECT (SEC,AREA,DSTDATA,BUCKET_SIZES,0) THEN
3208       IF SOURCE_DATA_BUCKET > DEF_CURRENT^.NUMBER THEN
3209           DEF_CURRENT^.NUMBER := SOURCE_DATA_BUCKET;
3210   IF FIND_OBJECT (SEC,AREA,DSTDATA,ALLOCATIONS,0) THEN
3211       DEF_CURRENT^.NUMBER := DEF_CURRENT^.NUMBER + SOURCE_DATA_ALLOC;
3212   IF FIND_OBJECT (SEC,AREA,DSTDATA,EXTENSIONS,0) THEN
3213       DEF_CURRENT^.NUMBER := DEF_CURRENT^.NUMBER + SOURCE_DATA_EXT;
3214   IF FIND_OBJECT (SEC,AREA,DSTIDX,BUCKET_SIZES,0) THEN
3215       IF SOURCE_INDEX_BUCKET > DEF_CURRENT^.NUMBER THEN
3216           DEF_CURRENT^.NUMBER := SOURCE_INDEX_BUCKET;
3217   IF FIND_OBJECT (SEC,AREA,DSTIDX,ALLOCATIONS,0) THEN
3218       DEF_CURRENT^.NUMBER := DEF_CURRENT^.NUMBER + SOURCE_INDEX_ALLOC;
3219   IF FIND_OBJECT (SEC,AREA,DSTIDX,EXTENSIONS,0) THEN
3220       DEF_CURRENT^.NUMBER := DEF_CURRENT^.NUMBER + SOURCE_INDEX_EXT;
3221   FOR KEYNUM := CURKEY TO MAXKEY DO
3222   BEGIN
```



```
3246
3247 { +
3248 Now point the key section(s) to the right areas.
3249 - }
3250 IF FIND_OBJECT (SEC,KEY,KEYNUM,DATA_AREA,0) THEN
3251     DEF_CURRENT^.NUMBER := DSTDATA;
3252
3253 IF FIND_OBJECT (SEC,KEY,KEYNUM,INDEX_AREA,0) THEN
3254     DEF_CURRENT^.NUMBER := DSTIDX;
3255
3256 IF FIND_OBJECT (SEC,KEY,KEYNUM,LEVEL1_INDEX_AREA,0) THEN
3257     DEF_CURRENT^.NUMBER := DSTIDX;
3258
3259
3260 END; { FOR }
3261
3262 { +
3263 Now get rid of the old area sections.
3264 - }
3265 IF SRCDATA <> DSTDATA THEN
3266     DELETE_PRIMARY_SECTION (AREA,SRCDATA);
3267
3268 IF SRCIDX <> DSTIDX THEN
3269     DELETE_PRIMARY_SECTION (AREA,SRCIDX);
3270
3271 END; { MERGE_AREA }
```

```
3277 ( ++
3278
3279 SHUFFLE_AREAS -- Implement Granularity.
3280
3281 This routine puts the area primary sections into their final state.
3282
3283 CALLING SEQUENCE:
3284
3285 SHUFFLE_AREAS;
3286
3287 INPUT PARAMETERS:
3288
3289 none
3290
3291 IMPLICIT INPUTS:
3292
3293 DEF_CURRENT
3294 DEF_HEAD
3295
3296 OUTPUT PARAMETERS:
3297
3298 none
3299
3300 IMPLICIT OUTPUTS:
3301
3302 DEF_CURRENT
3303 DEF_HEAD
3304
3305 ROUTINES CALLED:
3306
3307 none
3308
3309 ROUTINE VALUE:
3310
3311 none
3312
3313 SIGNALS:
3314
3315 none
3316
3317 SIDE EFFECTS:
3318
3319 none
3320
3321 -- }
```

```
3323 PROCEDURE SHUFFLE_AREAS;
3324
3325 VAR
3326     TEMP_KEY      : INTEGER;
3327     TEMP_AREA     : INTEGER;
3328     PROLOG_FOR_KEYS : INTEGER;
3329     PROLOG_FOR_AREAS : INTEGER;
3330
3331 BEGIN
3332     { +
3333     First, see what we have.
3334     - }
3335     SCAN_DEFINITION (TRUE);
3336
3337     { +
3338     You need at least 2 keys to support 3 or 4 areas.
3339     IF (
3340     (HIGH_KEY < 1)
3341     AND
3342     (IDATA[EDFSK_GROANULARITY] IN [ EDFSK_THREE, EDFSK_FOUR ])
3343     ) THEN
3344
3345         IDATA[EDFSK_GROANULARITY]      := EDFSK_TWO;
3346
3347     { +
3348     Now merge the areas according to whatever granularity was chosen.
3349     - }
3350     IF (
3351     (HIGH_KEY > 1)
3352     AND
3353     (IDATA[EDFSK_GROANULARITY] <> EDFSK_DOUBLE)
3354     ) THEN
3355
3356         BEGIN
3357
3358             TEMP_KEY      := HIGH_KEY;
3359
3360             { +
3361             Put all the alternate keys into areas 2 and 3.
3362             - }
3363             REPEAT
3364
3365                 TEMP_AREA  := TEMP_KEY * 2;
3366
3367                 MERGE_AREA (TEMP_KEY,TEMP_KEY,TEMP_AREA,2,(TEMP_AREA+1),3);
3368
3369                 TEMP_KEY    := TEMP_KEY - 1;
3370
3371             UNTIL TEMP_KEY < 2;
3372
3373         END;
3374
3375     CASE IDATA[EDFSK_GROANULARITY] OF
3376
3377         EDFSK_ONE :
```

```
3380 BEGIN
3381     IF HIGH_AREA > 1 THEN
3382         MERGE_AREA (1,HIGH_KEY,2,0,3,0);
3383     MERGE_AREA (0,HIGH_KEY,0,0,1,0);
3384 END;
3385 EDF$K_TWO :
3386 { +
3387 If we only have one key, then there's nothing to do.
3388 - }
3389 IF HIGH_KEY > 0 THEN
3390 BEGIN
3391     MERGE_AREA (1,HIGH_KEY,2,1,3,1);
3392 END;
3393 EDF$K_THREE :
3394 BEGIN
3395     MERGE_AREA (1,HIGH_KEY,2,2,3,2);
3396 END;
3397 EDF$K_FOUR :
3398 BEGIN
3399     { NULL-STATEMENT - all the work was done above } ;
3400 END;
3401 EDF$K_DOUBLE :
3402 BEGIN
3403     { NULL-STATEMENT - this is the initial situation } ;
3404 END;
3405 OTHERWISE
3406     { NULL-STATEMENT } ;
3407 END; { CASE }
3408 { +
3409 Lastly, add the length of the prolog to area 0.
3410 - }
3411 PROLOG_FOR_KEYS := 0;
```

```
3437 { +
3438 Key 0 descriptor is in the 1st prolog block. Others must go in following
3439 prolog blocks, 5 per block.
3440 - }
3441 IF HIGH_KEY > 0 THEN
3442 BEGIN
3443     PROLOG_FOR_KEYS      := HIGH_KEY DIV 5;
3444     IF ((HIGH_KEY MOD 5) <> 0) THEN
3445         PROLOG_FOR_KEYS  := PROLOG_FOR_KEYS + 1;
3446 END;
3447 { +
3448 Add in the key 0 descriptor.
3449 - }
3450 PROLOG_FOR_KEYS          := PROLOG_FOR_KEYS + 1;
3451 { +
3452 Prolog blocks for areas start after the prolog blocks for keys.
3453 No mixing is allowed. 7 area descriptors fit in a block.
3454 - }
3455 PROLOG_FOR_AREAS         := (HIGH_AREA+1) DIV 7;
3456 IF (((HIGH_AREA+1) MOD 7) <> 0) THEN
3457     PROLOG_FOR_AREAS     := PROLOG_FOR_AREAS + 1;
3458 { +
3459 Locate the line object that has the area 0 allocation in it
3460 and add the prolog size to it. If not found, let RMS default it all.
3461 - }
3462 IF FIND_OBJECT (SEC,AREA,0,ALLOCATIONS,0) THEN
3463     DEF_CURRENT^.NUMBER  := DEF_CURRENT^.NUMBER +
3464         MAX_FACTOR (IDATA[EDF$K CLUSTER_SIZE],
3465             (PROLOG_FOR_KEYS+PROLOG_FOR_AREAS),
3466             89); { = largest possible prolog }
3467 END; { SHUFFLE_AREAS }
```



```
3482 ( ++
3483
3484 CALC_ARRAY -- Do the calculations for a surface plot.
3485
3486 This routine sets up xy_array.
3487
3488 CALLING SEQUENCE:
3489
3490 CALC_ARRAY;
3491
3492 INPUT PARAMETERS:
3493
3494 none
3495
3496 IMPLICIT INPUTS:
3497
3498 none
3499
3500 OUTPUT PARAMETERS:
3501
3502 none
3503
3504 IMPLICIT OUTPUTS:
3505
3506 none
3507
3508 ROUTINES CALLED:
3509
3510 none
3511
3512 ROUTINE VALUE:
3513
3514 none
3515
3516 SIGNALS:
3517
3518 none
3519
3520 SIDE EFFECTS:
3521
3522 none
3523
3524 -- }
```

```
3526 PROCEDURE CALC_ARRAY;
3527
3528 VAR
3529     I           : INTEGER;
3530     J           : INTEGER;
3531     TEMP_INTEGER : INTEGER;
3532     TEMP_INT2    : INTEGER;
3533
3534 BEGIN
3535
3536     WRITELN (SHIFT, 'Working ...');
3537
3538     IF IDATA[EDFSK_SURFACE_OPTION] = EDFSK_FILL_SURFACE THEN
3539         GRAPH_TYPE      := EDFSC_SRF_DECREASING
3540     ELSE
3541         GRAPH_TYPE      := EDFSC_SRF_INCREASING;
3542
3543     CASE IDATA[EDFSK_SURFACE_OPTION] OF
3544         EDFSK_FILL_SURFACE :
3545             BEGIN
3546                 Y_LABEL      := 'Initial Load Fill Percent';
3547                 IDATA[EDFSK_DESIRED_FILL] := IDATA[EDFSK_Y_LOW];
3548             END;
3549         EDFSK_SIZE_SURFACE :
3550             BEGIN
3551                 IF VARIABLE_RECORDS THEN
3552                     Y_LABEL := 'Mean Record Size';
3553                 ELSE
3554                     Y_LABEL := 'Record Size';
3555                 IDATA[EDFSK_MEAN_RECORD_SIZE] := IDATA[EDFSK_Y_LOW];
3556             END;
3557         EDFSK_KEY_SURFACE :
3558             BEGIN
3559                 Y_LABEL      := 'Key Length';
3560                 IDATA[EDFSK_KEY_SIZE] := IDATA[EDFSK_Y_LOW];
3561             END;
3562         EDFSK_INIT_SURFACE :
3563     
```

```
3583 BEGIN
3584
3585     Y_LABEL := 'Initial Load Record Count';
3586     IDATA[EDFSK_INITIAL_COUNT] := IDATA[EDFSK_Y_LOW];
3587
3588 END;
3589
3590 EDFSK_ADDED_SURFACE :
3591
3592 BEGIN
3593
3594     Y_LABEL := 'Additional Record Count';
3595     IDATA[EDFSK_ADDED_COUNT] := IDATA[EDFSK_Y_LOW];
3596
3597 END;
3598
3599 OTHERWISE
3600
3601     { NULL-STATEMENT } ;
3602
3603 END; { CASE }
3604
3605 FOR I := 0 TO MAX_ARRAY_ROW DO
3606
3607 BEGIN
3608
3609     FOR J := 0 TO 31 DO
3610
3611 BEGIN
3612
3613     { +
3614     Bump the bucket size and recalculate.
3615     - }
3616     IDATA[EDFSK_BLOCKS_IN_BUCKET] := J + 1;
3617     XY_PLOT[I,J] := PROLOGUE3_DEPTH;
3618
3619 END; { FOR J }
3620
3621 { +
3622 Fill the color_row, and copy that into the array.
3623 - }
3624 TEMP_INTEGER := NATURAL_DEPTH;
3625
3626 FOR TEMP_INT2 := 0 TO 31 DO
3627
3628     COLOR_PLOT[I,TEMP_INT2] := COLOR_ROW[TEMP_INT2];
3629
3630 CASE IDATA[EDFSK_SURFACE_OPTION] OF
3631
3632     EDFSK_FILL_SURFACE : IDATA[EDFSK_DESIRED_FILL] :=
3633                         IDATA[EDFSK_DESIRED_FILL] + IDATA[EDFSK_Y_INCR];
3634
3635     EDFSK_SIZE_SURFACE : IDATA[EDFSK_MEAN_RECORD_SIZE] :=
3636                         IDATA[EDFSK_MEAN_RECORD_SIZE] + IDATA[EDFSK_Y_INCR];
3637
3638     EDFSK_KEY_SURFACE : IDATA[EDFSK_KEY_SIZE] :=
```

```
3640          IDATA[EDFSK_KEY_SIZE] + IDATA[EDFSK_Y_INCR];
3641
3642      EDFSK_INIT_SURFACE :      IDATA[EDFSK_INITIAL_COUNT] :=
3643          IDATA[EDFSK_INITIAL_COUNT] + IDATA[EDFSK_Y_INCR];
3644
3645      EDFSK_ADDED_SURFACE :      IDATA[EDFSK_ADDED_COUNT] :=
3646          IDATA[EDFSK_ADDED_COUNT] + IDATA[EDFSK_Y_INCR];
3647
3648      OTHERWISE
3649
3650          { NULL-STATEMENT } ;
3651
3652      END;      { CASE }
3653
3654      END;      { FOR I }
3655
3656      END;      { CALC_ARRAY }
```

```
3658 ( **
3659
3660 SETUP_GRAPH -- Setup to call EDF$GRAPH.
3661
3662 This routine sets up to call EDF$GRAPH.
3663
3664 CALLING SEQUENCE:
3665
3666 SETUP_GRAPH;
3667
3668 INPUT PARAMETERS:
3669
3670 none
3671
3672 IMPLICIT INPUTS:
3673
3674 none
3675
3676 OUTPUT PARAMETERS:
3677
3678 none
3679
3680 IMPLICIT OUTPUTS:
3681
3682 none
3683
3684 ROUTINES CALLED:
3685
3686 none
3687
3688 ROUTINE VALUE:
3689
3690 none
3691
3692 SIGNALS:
3693
3694 none
3695
3696 SIDE EFFECTS:
3697
3698 none
3699
3700 -- }
```



```
3702 PROCEDURE SETUP_GRAPH;
3703
3704 BEGIN
3705
3706   ( +
3707   Reset the boundary markers.
3708   - )
3709   IDATA[EDFSK_Y_LOW] := 0;
3710   IDATA[EDFSK_Y_HIGH] := 0;
3711   IDATA[EDFSK_Y_INCR] := 0;
3712
3713   IF NOT AUTO_TUNE THEN
3714     WRITELN;
3715
3716   ( +
3717   Now fill up the xy_array (if needed).
3718   - )
3719   IF IDATA[EDFSK_SURFACE_OPTION] = EDFSK_INIT_SURFACE THEN
3720     BEGIN
3721       QUERY (EDFSK_INITIAL_COUNT_LOW);
3722       QUERY (EDFSK_INITIAL_COUNT_HIGH);
3723       AUTO_SCALE (0,EDFSK_1GIGA);
3724     END
3725   ELSE
3726     QUERY (EDFSK_INITIAL_COUNT);
3727
3728     QUERY (EDFSK_LOAD_METHOD);
3729     QUERY (EDFSK_ASCENDING_LOAD);
3730
3731     IF IDATA[EDFSK_SURFACE_OPTION] = EDFSK_ADDED_SURFACE THEN
3732       BEGIN
3733         QUERY (EDFSK_ADDED_COUNT_LOW);
3734         QUERY (EDFSK_ADDED_COUNT_HIGH);
3735         AUTO_SCALE (0,EDFSK_1GIGA);
3736       END
3737     ELSE
3738       QUERY (EDFSK_ADDED_COUNT);
3739
3740       QUERY (EDFSK_ASCENDING_ADDED);
3741       QUERY (EDFSK_KEY_DIST);
3742
3743       IF IDATA[EDFSK_SURFACE_OPTION] = EDFSK_FILL_SURFACE THEN
3744         BEGIN
3745           QUERY (EDFSK_FILL_LOW);
3746         END
3747       ELSE
3748         QUERY (EDFSK_ADDED_COUNT);
3749
3750       QUERY (EDFSK_ASCENDING_ADDED);
3751       QUERY (EDFSK_KEY_DIST);
3752
3753       IF IDATA[EDFSK_SURFACE_OPTION] = EDFSK_FILL_SURFACE THEN
3754         BEGIN
3755           QUERY (EDFSK_FILL_LOW);
3756         END
3757       ELSE
3758         QUERY (EDFSK_ADDED_COUNT);
```

```
3759     QUERY (EDFSK_FILL_HIGH);
3760     AUTO_SCALE (31,100);
3761
3762 END
3763
3764 ELSE
3765     QUERY (EDFSK_DESIRED_FILL);
3766
3767     QUERY (EDFSK_RECORD_FORMAT);
3768
3769     IF IDATA[EDFSK_SURFACE_OPTION] = EDFSK_SIZE_SURFACE THEN
3770
3771     BEGIN
3772
3773         QUERY (EDFSK_SIZE_LOW);
3774         QUERY (EDFSK_SIZE_HIGH);
3775         AUTO_SCALE (T,CUR_MAX_REC);
3776         IDATA[EDFSK_MAX_RECORD_SIZE] := IDATA[EDFSK_Y_HIGH];
3777
3778     END
3779
3780 ELSE
3781
3782     ASK_MEAN_RECORD_SIZE;
3783
3784     QUERY (EDFSK_KEY_TYPE);
3785     QUERY (EDFSK_SEGMENTED);
3786     SEGMENT_NUMBER := 0;
3787
3788     IF IDATA[EDFSK_SURFACE_OPTION] = EDFSK_KEY_SURFACE THEN
3789
3790     BEGIN
3791
3792         QUERY (EDFSK_KEY_LOW);
3793         QUERY (EDFSK_KEY_HIGH);
3794         AUTO_SCALE (T,MAX_KEY_SIZE);
3795
3796     END
3797
3798 ELSE
3799
3800     ASK_KEY_SIZE;
3801
3802     ASK_KEY_POSITION;
3803     ASK_KEY_DUPS;
3804     QUERY (EDFSK_PROLOGUE_VERSION);
3805     ASK_KEY_COMP;
3806     ASK_REC_COMP;
3807     ASK_IDX_COMP;
3808
3809     IF NOT AUTO_TUNE THEN
3810
3811         WRITELN;
3812
3813         ( +
3814         Since calc_array is called only if it's not a line plot, we don't
3815
```

```
3816 have to conditionalize its writes for not auto_tune. (nointeractive
3817 uses only line plots)
3818 - )
3819 IF IDATA[EDFSK_SURFACE_OPTION] <> EDFSK_LINE_SURFACE THEN
3820
3821 ( +
3822 Now fill the xy_array (if needed).
3823 - )
3824 CALC_ARRAY;
3825
3826 END; ( SETUP_GRAPH )
```

```
3828 { ++
3829
3830 PLOT_AND_DESIGN -- Show the graph on the screen and design the file.
3831
3832 This routine displays the graph for the file and lets the user change
3833 the file parameters (design the file).
3834
3835 CALLING SEQUENCE:
3836
3837 PLOT_AND_DESIGN;
3838
3839 INPUT PARAMETERS:
3840
3841 none
3842
3843 IMPLICIT INPUTS:
3844
3845 CONTROL_ZEE_Typed
3846 SYSS$INPUT:
3847
3848 OUTPUT PARAMETERS:
3849
3850 none
3851
3852 IMPLICIT OUTPUTS:
3853
3854 CONTROL_ZEE_Typed
3855 SYSS$OUTPUT:
3856
3857 ROUTINES CALLED:
3858
3859 QUERY (EDF$K_SURFACE_OPTION)
3860 SETUP_GRAPH
3861
3862 ROUTINE VALUE:
3863
3864 none
3865
3866 SIGNALS:
3867
3868 none
3869
3870 SIDE EFFECTS:
3871
3872 none
3873
3874 -- }
```

3876 PROCEDURE PLOT_AND_DESIGN;

3877 BEGIN

3878 { +
3879 See what kind of graph he wants.3880 - }
3881 QUERY (EDF\$K_SURFACE_OPTION);3882 { +
3883 Find out what the user's parameters are, and fill the xy_array (if needed).
3884 Indicate that questions should be visible now - even if optimizing.3885 - }
3886 SETUP_GRAPH;
3887 VISIBCE_QUESTION := TRUE;
3888 TAKE_DEFAULTS := AUTO_TUNE;3889 { +
3890 Make bottom lines of screen scroll.3891 - }
3892 LIB\$SET_SCROLL (PROMPT_LINE,LINES_PER_PAGE);
3893 SCROLLING_SET := TRUE;
3894 WAIT_HELP := TRUE;3895 { +
3896 Init to do non-move on 1st time thru
3897 - }
3898 FIRST_PLOT := TRUE;3899 { +
3900 Show the user the calculated depths.3901 - }
3902 PLOT_GRAPH;3903 { +
3904 This will loop until the user types control/Z or
3905 LINK_RESULTS makes LINKED true.3906 - }
3907 LINKED := FALSE;

3908 WHILE NOT LINKED DO

3909 BEGIN

3910 { +
3911 See what the user wants to vary.3912 - }
3913 QUERY (EDF\$K_DESIGN_CYCLE);

3914 CASE [DATA[EDF\$K_DESIGN_CYCLE] OF

3915 EDF\$K_RF : QUERY (EDF\$K_RECORD_FORMAT);

3916 EDF\$K_RS : ASK_MEAN_RECORD_SIZE;

3917 EDF\$K_KL : ASK_KEY_SIZE;

3918


```
EDFSK_BF :      QUERY (EDFSK_DESIRED_FILL);
EDFSK_EM :      QUERY (EDFSK_BUCKET_WEIGHT);
EDFSK_IL :      QUERY (EDFSK_INITIAL_COUNT);
EDFSK_KP :      ASK_KEY_POSITION;
EDFSK_LM :      QUERY (EDFSK_LOAD_METHOD);
EDFSK_AR :      QUERY (EDFSK_ADDED_COUNT);
EDFSK_DK :      ASK_KEY_DUPS;
EDFSK_RC :      ASK_REC_COMP;
EDFSK_KC :      ASK_KEY_COMP;
EDFSK_IC :      ASK_IDX_COMP;
EDFSK_PV :      QUERY (EDFSK_PROLOGUE_VERSION);
EDFSK_KT :      QUERY (EDFSK_KEY_TYPE);
EDFSK_FINIS :   LINK_RESULTS;
EDFSK_WP :
BEGIN
    ( +
    This is the write fresh plot function.
    - )
    FIRST_PLOT      := TRUE;
    PLOT_GRAPH;
END;
OTHERWISE
    { NULL-STATEMENT } ;
END;    { CASE }
( +
If we just finished putting up a new plot, or we're done,
don't do it again.
- )
IF NOT ((IDATA[EDFSK_DESIGN_CYCLE] = EDFSK_WP) OR LINKED) THEN
BEGIN
    IF IDATA[EDFSK_SURFACE_OPTION] <> EDFSK_LINE_SURFACE THEN
        CALC_ARRAY;
    PLOT_GRAPH;
```

EDFDESIGN
V04-000

Source Listing

C 10
16-Sep-1984 01:10:30
5-Sep-1984 13:36:36

VAX-11 Pascal V2.4-277
DISK\$VMSMASTER:[EDF.SRC]EDFDESIGN.PAS;1 (34)

Page 81

3990
3991
3992
3993
3994
3995
3996
3997

```
      END;      { IF IDATA[EDF$K_DESIGN_CYCLE] <> EDF$K_WP }  
    END;      { WHILE }  
    EDF$RESET_SCROLL;  
  END;      { PLOT_AND_DESIGN }
```

```
3999      { ++
4000
4001      SEQ_REL_WORK -- Do the calculations for designing Seq and Rel files.
4002
4003      This routine does all the work.
4004
4005      CALLING SEQUENCE:
4006
4007      SEQ_REL_WORK;
4008
4009      INPUT PARAMETERS:
4010
4011      none
4012
4013      IMPLICIT INPUTS:
4014
4015      none
4016
4017      OUTPUT PARAMETERS:
4018
4019      none
4020
4021      IMPLICIT OUTPUTS:
4022
4023      none
4024
4025      ROUTINES CALLED:
4026
4027      none
4028
4029      ROUTINE VALUE:
4030
4031      none
4032
4033      SIGNALS:
4034
4035      none
4036
4037      SIDE EFFECTS:
4038
4039      none
4040
4041      -- }
```

```
4043 PROCEDURE SEQ_REL_WORK;  
4044  
4045 BEGIN  
4046     ( +  
4047     Find out how the user is going to use the file.  
4048     - )  
4049     QUERY (EDFSK_NUMBER_RECORDS);  
4050     QUERY (EDFSK_RECORD_FORMAT);  
4051     QUERY (EDFSK_BLOCK_SPAN);  
4052     ASK_MEAN_RECORD_SIZE;  
4053  
4054     ( +  
4055     Stuff the definition.  
4056     - )  
4057     INIT_DEF;  
4058     NON_REY_DEF;  
4059  
4060 END;    { SEQ_REL_WORK }  
4061
```

```
4063 { ++
4064
4065 INDEXED_DESIGN -- Do the dirty work to design an indexed file.
4066
4067 This routine does all the calculations needed to design an indexed file.
4068 It also serves the redesign and optimize functions.
4069
4070 CALLING SEQUENCE:
4071
4072 INDEXED_DESIGN (REDESIGN_FLAG,ADD_KEY_FLAG);
4073
4074 INPUT PARAMETERS:
4075
4076 REDESIGN_FLAG
4077 ADD_KEY_FLAG
4078
4079 IMPLICIT INPUTS:
4080
4081 OPTIMIZING
4082 CONTROL_ZEE_TYPED
4083 SYSSINPUT:
4084
4085 OUTPUT PARAMETERS:
4086
4087 none
4088
4089 IMPLICIT OUTPUTS:
4090
4091 CONTROL_ZEE_TYPED
4092 SYSSOUTPUT:
4093
4094 ROUTINES CALLED:
4095
4096 PLOT_AND_DESIGN
4097
4098 ROUTINE VALUE:
4099
4100 none
4101
4102 SIGNALS:
4103
4104 none
4105
4106 SIDE EFFECTS:
4107
4108 none
4109
4110 -- }
```

```
4112 PROCEDURE INDEXED_DESIGN (REDESIGN_FLAG, ADD_KEY_FLAG : BOOLEAN);
4113
4114 VAR
4115     BEGINING_KEY      : INTEGER;
4116     ENDING_KEY         : INTEGER;
4117     ACTIVE_KEY_INDEX  : INTEGER;
4118
4119 BEGIN
4120
4121     { +
4122     Find out the cluster factor of the target disk.
4123     - }
4124     QUERY (EDFSK_CLUSTER_SIZE);
4125
4126     { +
4127     Initialize the script.
4128     - }
4129     IF NOT OPTIMIZING THEN
4130     BEGIN
4131         IF REDESIGN_FLAG THEN
4132         BEGIN
4133             { +
4134             The add_key script has already setup [active_key].
4135             - }
4136             IF NOT ADD_KEY_FLAG THEN
4137             BEGIN
4138                 QUERY (EDFSK_ACTIVE_KEY);
4139
4140                 BEGINING_KEY      := IDATA[EDFSK_ACTIVE_KEY];
4141                 ENDING_KEY         := BEGINING_KEY;
4142             END
4143         ELSE
4144         BEGIN
4145             QUERY (EDFSK_NUMBER_KEYS);
4146             BEGINING_KEY      := 0;
4147             ENDING_KEY         := IDATA[EDFSK_NUMBER_KEYS] - 1;
4148         END;
4149     END { IF TRUE NOT OPTIMIZING }
4150 ELSE
4151 BEGIN
4152     SCAN_DEFINITION (TRUE);
4153     IDATA[EDFSK_NUMBER_KEYS] := HIGH_KEY + 1;
4154     BEGINING_KEY      := 0;
4155     ENDING_KEY         := HIGH_KEY;
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
```



```
4169
4170     END:      ( IF FALSE NOT OPTIMIZING )
4171
4172     ( +
4173     Now loop until all his keys are (re)defined.
4174     - )
4175     FOR ACTIVE_KEY_INDEX := BEGINING_KEY TO ENDING_KEY DO
4176
4177     BEGIN
4178
4179         IDATA[EDF$K_ACTIVE_KEY] := ACTIVE_KEY_INDEX;
4180
4181         IF (
4182             (REDESIGN_FLAG)
4183             AND
4184             (NOT ADD_KEY_FLAG)
4185         ) THEN
4186
4187             WARN_OF_ERASE;
4188
4189             PLOT_AND_DESIGN;
4190
4191     END:      ( FOR ... )
4192
4193     ( +
4194     Now that we're done with the hard part, set the function default
4195     to succeed.
4196     - )
4197     IF AUTO_TUNE THEN
4198
4199         QTAB[EDF$K_CURRENT_FUNCTION].DEFAULT      := EDF$K_EXIT;
4200
4201     END:      ( INDEXED_DESIGN )
4202
4203     END.
4204     ( End of file: SRC$:EDFDESIGN.PAS )
```

```
65 72 47 20 66 6F 20 65 6C 69 46 20 41 20
49 20 31 33 20 6E 61 68 74 20 72 65 74 61
61 68 20 73 6C 65 76 65 4C 20 78 65 64 6E
69 66 69 63 65 70 73 20 6E 65 65 62 20 73
20 20 20 68 74 70 65 44 20 78 65 64 6E 49
20 20 20 20 20 20 20 20 20 20 20 20 20
00 00 00 6C 35 3F 58 1B
00000000 00000680 02C00000 00000000 04080004
00000000 0000000C 00000000 00000020 0000000D
00000000 00000680 02C00000 00000000 04080004
00000000 0000000C 00000000 00000020 0000000D
00000000 0000001F
6E 6F 69 74 69 6E 69 66 65 44 20 65 68 54
00 00 00 79 65 48 20 66 6F 20
61 6C 70 65 72 20 65 62 20 6C 6C 69 77 20
00 00 2E 64 65 63
44 20 74 6E 65 72 72 75 43 20 65 68 54 20
6C 6C 69 77 20 6E 6F 69 74 69 6E 69 66 65
20 2E 64 65 63 61 6C 70 65 72 20 65 62 20
00 00
00000000 00100F00 00000000 00000000
00 00 00
4B 20 66 6F 20 68 74 70 65 44 20 65 68 54
20 64 65 74 61 6D 69 74 73 45 20 73 69 20
74 61 65 72 47 20 6F 4E 20 65 62 20 6F 74
00 00 72 65
2C 73 6C 65 76 65 6C 20 78 65 64 6E 49 20
20 73 69 20 68 63 69 68 77 20
2E 73 6C 65 76 65 6C 20 6C 61 74 6F 54 20
00 00
46 20 00 2E 2E 2E 20 67 6E 69 68 72 6F 57
20 20 64 61 6F 4C 20 6C 61 69 74 69 6E 49
20 20 20 74 6E 65 63 72 65 50 20 6C 6C 69
20 20 20 20 20 20 20 20 20 20 20 20 20
69 53 20 64 72 6F 63 65 52 20 20 6E 61 65 4D
20 20 20 20 20 20 20 20 20 20 20 20 65 7A
20 20 20 65 7A 69 53 20 64 72 6F 63 65 52
20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 68 74 67 6E 65 4C 20 79 65 48
20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20
52 20 64 61 6F 4C 20 6C 61 69 74 69 6E 49
20 20 20 74 6E 75 6F 43 20 64 72 6F 63 65
20 20 20 20 20 20 20 20
```

```
.TITLE EDFDESIGN
.IDENT \V04-000\

00000 .PSECT $CODE,PIC,CON,REL,LCL,SHR,EXE,RD,NOWRT,2
00000 C.AAA: .ASCII \ A File of Greater than 31 Index Levels \-
0000E \has been specified. \
0001C
0002A
00038
0003C C.AAB: .ASCII \Index Depth \
0004A
00058
0005C C.AAC: .ASCII <27>\[?5\[\<0><0><0>
00064 C.AAD: .LONG ^X4080004,0,^X2C00000,^X680,0,^XD,^X20,0,-
00078 ^XC,0,^X1F
0008C
00090 C.AAE: .LONG ^X4080004,0,^X2C00000,^X680,0,^XD,^X20,0,-
000A4 ^XC,0,^X1F
000B8
000BC C.AAF: .ASCII \The Definition of Key\<0><0><0>
000CA
000D4 C.AAG: .ASCII \ will be replaced.\<0><0>
000E2
000E8 C.AAH: .ASCII \ The Current Definition will be replaced\
000F6 \. \<0><0>
00104
00112
00114 C.AAI: .LONG 0,0,^X100F00,0
00124 .BYTE 0,0,0
00127 .BLKB 1
00128 C.AAJ: .ASCII \The Depth of Key\
00136
00138 C.AAK: .ASCII \ is Estimated to be No Greater\<0><0>
00146
0C154
00158 C.AAL: .ASCII \than \<0><0><0>
00160 C.AAM: .ASCII \ Index levels, which is \
0016E
00178 C.AAN: .ASCII \ Total levels.\<0><0>
00186
00188 C.AAO: .ASCII \Working ... \<0>
00194 C.AAP: .ASCII \Initial Load Fill Percent \
001A2
001B0
001B4 C.AAQ: .ASCII \Mean Record Size \
001C2
001D0
001D4 C.AAR: .ASCII \Record Size \
001E2
001F0
001F4 C.AAS: .ASCII \Key Length \
00202
00210
00214 C.AAT: .ASCII \Initial Load Record Count \
00222
00230
```

| | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 63 | 65 | 52 | 20 | 6C | 61 | 6E | 6F | 69 | 74 | 69 | 64 | 64 | 41 |
| 20 | 20 | 20 | 20 | 20 | 74 | 6E | 75 | 6F | 43 | 20 | 64 | 72 | 6F |
| | | | | | | | | | | 20 | 20 | 20 | 20 |

00234 C.AAU: .ASCII \Additional Record Count
00242
00250

00000 PROLOGUE3 BUCKETS:

: 0174

| | | | | | | | | |
|-----------|-----------|-----------|-----|------|-------|--------|--------------------------------------|--------|
| | 5E | | 0C | 07FC | 00000 | WORD | *M<R2,R3,R4,R5,R6,R7,R8,R9,R10> | |
| | 52 | 04 | BC | C2 | 00002 | SUBL2 | #12,SP | |
| | 53 | 08 | BC | D0 | 00005 | MOVL | @4(R12),INIT_NUMBER_RECORDS | |
| | 5C | 0C | BC | D0 | 00009 | MOVL | @8(R12),ADDED_NUMBER_RECORDS | |
| FC | AD | | 5C | D0 | 0000D | MOVL | @12(R12),INDEX_LEVEL | |
| | | FC | AD | 9F | 00015 | MOVL | INDEX_LEVEL,-4(FP) | : 0199 |
| 00000000G | EF | | 01 | FB | 00018 | PUSHAB | -4(FP) | |
| | 54 | | 50 | D0 | 0001F | CALLS | #1,CALC_BUC_OVERHEAD | |
| FC | AD | | 5C | D0 | 00022 | MOVL | R0,BUCKET_OVERHEAD | |
| | | FC | AD | 9F | 00026 | MOVL | INDEX_LEVEL,-4(FP) | : 0200 |
| 00000000G | EF | | 01 | FB | 00029 | PUSHAB | -4(FP) | |
| 51 | 000000D8G | | EF | 4E | 00030 | CALLS | #1,CALC_REC_OVERHEAD | |
| | | | 5C | D5 | 00037 | CVTLF | IDATA+2T6,RT | : 0202 |
| | | | 00V | 12 | 00039 | TSTL | INDEX_LEVEL | |
| 55 | 51 | 00000020G | EF | 45 | 0003B | BNEQ | 8\$ | |
| | 55 | 00000084G | 55 | 4A | 00043 | MULF3 | RDATA+32,R1,R5 | : 0206 |
| | | | EF | D5 | 00046 | CVTFL | R5,R5 | |
| | | | 00V | 12 | 0004C | TSTL | IDATA+132 | |
| | 56 | | 55 | D0 | 0004E | BNEQ | 3\$ | |
| 57 | 000000E8G | | EF | C3 | 00051 | MOVL | R5,KEY_SAVINGS | : 0218 |
| | 57 | 000000D8G | EF | 4E | 0005D | SUBL3 | IDATA+216,IDATA+232,R7 | : 0220 |
| | 57 | 00000024G | 57 | 44 | 00060 | CVTLF | R7,R7 | |
| | 57 | | EF | 4A | 00067 | MULF2 | RDATA+36,R7 | |
| | 57 | | 57 | C0 | 0006A | CVTFL | R7,DATA_SAVINGS | |
| 57 | 000000E8G | | 56 | C3 | 0006D | ADDL2 | KEY_SAVINGS,DATA_SAVINGS | : 0227 |
| | EF | | 57 | C3 | 0006D | SUBL3 | DATA_SAVINGS,IDATA+232,RECORD_SIZE | |
| | | | 00V | 11 | 00075 | BRB | 9\$ | |
| 56 | 000000D8G | | 55 | C3 | 00077 | SUBL3 | INDEX_SAVINGS,IDATA+216,TEMP_REC | : 0243 |
| 58 | 000000ECG | | 55 | C5 | 0007F | SUBL3 | #9,IDATA+236,R8 | : 0245 |
| | 56 | | 58 | C0 | 00087 | MULL3 | R8,TEMP_REC | |
| 58 | 000000ECG | | 01 | C1 | 0008A | ADDL2 | #1,IDATA+236,R8 | : 0248 |
| 57 | 56 | | 58 | C7 | 00092 | ADDL3 | R8,TEMP_REC,RECORD_SIZE | |
| 59 | 00 | | 00 | 7A | 00096 | DIVL3 | R8,TEMP_REC,RECORD_SIZE | : 0250 |
| 59 | 59 | | 58 | 7B | 0009B | EMUL | #0,#0,TEMP_REC,R9 | |
| | | | 59 | D5 | 000A0 | EDIV | R8,R9,R9,R9 | |
| | | | 00V | 18 | 000A2 | TSTL | R9 | |
| | 59 | | 58 | C0 | 000A4 | BGEQ | 4\$ | |
| | | | 59 | D5 | 000A7 | ADDL2 | R8,R9 | |
| | | | 00V | 13 | 000A9 | TSTL | R9 | |
| | | | 57 | D6 | 000AB | BEQL | 9\$ | |
| | | | 00V | 11 | 000AD | INCL | RECORD_SIZE | : 0252 |
| | 51 | 00000028G | EF | 44 | 000AF | BRB | 9\$ | |
| | 55 | | 51 | 4A | 000B6 | MULF2 | RDATA+40,R1 | : 0269 |
| 57 | 000000D8G | | 55 | C3 | 000B9 | CVTFL | R1,INDEX_SAVINGS | |
| 54 | 00000000G | | 54 | C3 | 000C1 | SUBL3 | INDEX_SAVINGS,IDATA+216,RECORD_SIZE | : 0271 |
| | | | 54 | 4E | 000C9 | SUBL3 | BUCKET_OVERHEAD,BYTES_PER_BUCKET,R4 | : 0283 |
| | 54 | 00000004G | EF | 45 | 000CC | CVTLF | R4,R4 | |
| | 55 | | 55 | 4A | 000D4 | MULF3 | RDATA+4,R4,R5 | |
| | 54 | 00000000G | EF | 44 | 000D7 | CVTFL | R5,INIT_AVAILABLE_BYTES | |
| | 54 | | 54 | 4A | 000DE | MULF2 | RDATA,R4 | : 0285 |
| | 50 | | 57 | C0 | 000E1 | CVTFL | R4,ADDED_AVAILABLE_BYTES | |
| | 55 | | 50 | C6 | 000E4 | ADDL2 | RECORD_SIZE,RECORD_OVERHEAD | : 0292 |
| | | | | | | DIVL2 | RECORD_OVERHEAD,INIT_AVAILABLE_BYTES | |

| | | | | | | | | |
|----|----|---------------|---------------|----|-------|-------|---|--------|
| | | 54 | 50 | C6 | 000E7 | DIVL2 | R0,ADDED_AVAILABLE_BYTES | : 0294 |
| | | | 5C | D5 | 000EA | TSTL | INDEX_LEVEL | : 0301 |
| | | | 00V | 12 | 000EC | BNEQ | 12\$ | |
| | | 01 | 55 | D1 | 000EE | CMPL | INIT_RECORDS_PER_BUCKET,#1 | |
| | | | 00V | 18 | 000F1 | BGEQ | 12\$ | |
| | | 55 | 01 | D0 | 000F3 | MOVL | #1,INIT_RECORDS_PER_BUCKET | : 0303 |
| | | | 00V | 11 | 000F6 | BRB | 16\$ | |
| | | | 5C | D5 | 000F8 | TSTL | INDEX_LEVEL | : 0305 |
| | | | 00V | 15 | 000FA | BLEQ | 16\$ | |
| | | 02 | 55 | D1 | 000FC | CMPL | INIT_RECORDS_PER_BUCKET,#2 | |
| | | | 00V | 18 | 000FF | BGEQ | 16\$ | |
| | | 55 | 02 | D0 | 00101 | MOVL | #2,INIT_RECORDS_PER_BUCKET | : 0307 |
| | | | 5C | D5 | 00104 | TSTL | INDEX_LEVEL | : 0309 |
| | | | 00V | 12 | 00106 | BNEQ | 19\$ | |
| | | 01 | 54 | D1 | 00108 | CMPL | ADDED_RECORDS_PER_BUCKET,#1 | |
| | | | 00V | 18 | 0010B | BGEQ | 19\$ | |
| | | 54 | 01 | D0 | 0010D | MOVL | #1,ADDED_RECORDS_PER_BUCKET | : 0311 |
| | | | 00V | 11 | 00110 | BRB | 23\$ | |
| | | | 5C | D5 | 00112 | TSTL | INDEX_LEVEL | : 0313 |
| | | | 00V | 15 | 00114 | BLEQ | 23\$ | |
| | | 02 | 54 | D1 | 00116 | CMPL | ADDED_RECORDS_PER_BUCKET,#2 | |
| | | | 00V | 18 | 00119 | BGEQ | 23\$ | |
| | | 54 | 02 | D0 | 0011B | MOVL | #2,ADDED_RECORDS PER BUCKET | : 0315 |
| | | 55 | 54 | C1 | 0011E | ADDL3 | ADDED RECORDS PER BUCKET,- | : 0319 |
| | | | | | 00127 | | INIT_RECORDS PER BUCKET,- | |
| | | | | | 00127 | | RECS_PER_BUCKET[INDEX_LEVEL] | |
| | | 57 | 00000000GEF4C | DE | 00127 | MOVAL | INIT_NUMBER_BUCKETS[INDEX_LEVEL],R7 | : 0325 |
| | 67 | 52 | 55 | C7 | 0012F | DIVL3 | INIT_RECORDS PER BUCKET,- | |
| | | | | | 00133 | | INIT_NUMBER_RECORDS,(R7) | |
| | | 59 | 00000000GEF4C | DE | 00133 | MOVAL | ADDED_NUMBER_BUCKETS[INDEX_LEVEL],R9 | : 0327 |
| | 69 | 53 | 54 | C7 | 0013B | DIVL3 | ADDED_RECORDS PER BUCKET,- | |
| | | | | | 0013F | | ADDED_NUMBER_RECORDS,(R9) | |
| 50 | 52 | 00 | 00 | 7A | 0013F | EMUL | #0,#0,INIT NUMBER RECORDS,R0 | : 0333 |
| 50 | 50 | 50 | 55 | 7B | 00144 | EDIV | INIT_RECORDS_PER_BUCKET,R0,R0,R0 | |
| | | | 50 | D5 | 00149 | TSTL | R0 | |
| | | | 00V | 18 | 0014B | BGEQ | 24\$ | |
| | | 50 | 55 | C0 | 0014D | ADDL2 | INIT_RECORDS_PER_BUCKET,R0 | |
| | | | 50 | D5 | 00150 | TSTL | R0 | |
| | | | 00V | 13 | 00152 | BEQL | 26\$ | |
| | | | 67 | D6 | 00154 | INCL | (R7) | : 0335 |
| 50 | 53 | 00 | 00 | 7A | 00156 | EMUL | #0,#0,ADDED NUMBER RECORDS,R0 | : 0338 |
| 50 | 50 | 50 | 54 | 7B | 0015B | EDIV | ADDED_RECORDS_PER_BUCKET,R0,R0,R0 | |
| | | | 50 | D5 | 00160 | TSTL | R0 | |
| | | | 00V | 18 | 00162 | BGEQ | 27\$ | |
| | | 50 | 54 | C0 | 00164 | ADDL2 | ADDED_RECORDS_PER_BUCKET,R0 | |
| | | | 50 | D5 | 00167 | TSTL | R0 | |
| | | | 00V | 13 | 00169 | BEQL | 29\$ | |
| | | | 69 | D6 | 0016B | INCL | (R9) | : 0340 |
| | | | EF | D5 | 0016D | TSTL | [DATA+132 | : 0347 |
| | | | 00V | 12 | 00173 | BNEQ | 31\$ | |
| | | 00000000GEF4C | 67 | D0 | 00175 | MOVL | (R7),INIT PRIMARY BUCKETS[INDEX_LEVEL] | : 0351 |
| | | 00000000GEF4C | 69 | D0 | 0017D | MOVL | (R9),ADDED PRIMARY BUCKETS[INDEX_LEVEL] | : 0353 |
| | | 00000000G EF | 5C | D0 | 00185 | MOVL | INDEX_LEVEL,DEEPEST | : 0361 |
| | | 01 | 69 | D1 | 0018C | CMPL | (R9),#1 | : 0367 |
| | | | 00V | 14 | 0018F | BGTR | 34\$ | |
| | | | 5C | D5 | 00191 | TSTL | INDEX_LEVEL | |
| | | | 00V | 13 | 00193 | BEQL | 34\$ | |

| | | | | | | | |
|--------------|-----------|-----------|-------|------------|-------------|-------------------------------|----------------------------------|
| 01 | 67 | D1 | 00195 | CMPL | (R7),#1 | | |
| | 03 | 14 | 00198 | BGTR | ,+3 | | |
| | 0000V | 31 | 0019A | BRW | 45\$ | | |
| | 5C | D5 | 0019D | 34\$: TSTL | INDEX_LEVEL | : 0380 | |
| | 00V | 12 | 0019F | BNEQ | 41\$ | | |
| | 54 | 94 | 001A1 | CLRB | FOUND | : 0384 | |
| 00V00000000G | EF | 00 | E1 | 001A3 | BBC | #0,OPTIMIZING,37\$: 0386 | |
| 000000000G | EF | 00 | FB | 001AB | CALLS | #0,POINT_AT_ANALYSIS : 0390 | |
| | 00000000 | 8F | DF | 001B2 | PUSHAL | #0 : 0392 | |
| | 15 | 8F | 9F | 001B8 | PUSHAB | #21 | |
| | 00000084G | EF | 9F | 001BB | PUSHAB | IDATA+132 | |
| | 04 | 8F | 9F | 001C1 | PUSHAB | #4 | |
| | 01 | 8F | 9F | 001C4 | PUSHAB | #1 | |
| 00000000G | EF | 05 | FB | 001C7 | CALLS | #5,FIND_OBJECT | |
| | 54 | 50 | 90 | 001CE | MOVB | R0,FOUND | |
| 00000000G | EF | 00 | FB | 001D1 | CALLS | #0,POINT_AT_DEFINITION : 0396 | |
| | 00V | 54 | E9 | 001D8 | 37\$: BLBC | FOUND,39\$: 0400 | |
| | 54 | 00000000G | EF | D0 | 001DB | MOVL | DEF_CURRENT,R4 : 0404 |
| | 52 | 27 | A4 | D0 | 001E2 | MOVL | 39(R4),INIT_NUMBER_RECORDS |
| | | | 00V | 11 | 001E6 | BRB | 42\$ |
| | 52 | | 67 | D0 | 001E8 | 39\$: MOVL | (R7),INIT_NUMBER_RECORDS : 0412 |
| | | | 00V | 11 | 001EB | BRB | 42\$ |
| | 52 | | 67 | D0 | 001ED | 41\$: MOVL | (R7),INIT_NUMBER_RECORDS : 0422 |
| | 53 | | 69 | D0 | 001F0 | 42\$: MOVL | (R9),ADDED_NUMBER_RECORDS : 0426 |
| | | | 5C | D6 | 001F3 | INCL | INDEX_LEVEL : 0428 |
| | 1F | | 5C | D1 | 001F5 | CMPL | INDEX_LEVEL,#31 : 0433 |
| | | | 03 | 14 | 001F8 | BGTR | ,+3 |
| | | 0000V | 31 | 001FA | BRW | 44\$ | |
| | 00000000G | EF | 9F | 001FD | PUSHAB | SHIFT : 0437 | |
| | | 04 | DD | 00203 | PUSHL | #4 | |
| | 00000000G | EF | 9F | 00205 | PUSHAB | PASS\$V_OUTPUT | |
| 00000000G | EF | 03 | FB | 0020B | CALLS | #3,PASS\$WRITE_STRING | |
| | 00000000G | EF | 9F | 00212 | PUSHAB | ANSI_REVERSE | |
| | | 04 | DD | 00218 | PUSHL | #4 | |
| | 00000000G | EF | 9F | 0021A | PUSHAB | PASS\$V_OUTPUT | |
| 00000000G | EF | 03 | FB | 00220 | CALLS | #3,PASS\$WRITE_STRING | |
| | FFFFFB7F | EF | 9F | 00227 | PUSHAB | C.AAA | |
| | | 3C | DD | 0022D | PUSHL | #60 | |
| | 00000000G | EF | 9F | 0022F | PUSHAB | PASS\$V_OUTPUT | |
| 00000000G | EF | 03 | FB | 00235 | CALLS | #3,PASS\$WRITE_STRING | |
| | 00000000G | EF | 9F | 0023C | PUSHAB | ANSI_RESET | |
| | | 04 | DD | 00242 | PUSHL | #4 | |
| | 00000000G | EF | 9F | 00244 | PUSHAB | PASS\$V_OUTPUT | |
| 00000000G | EF | 03 | FB | 0024A | CALLS | #3,PASS\$WRITE_STRING | |
| | 00000000G | EF | 9F | 00251 | PUSHAB | PASS\$V_OUTPUT | |
| 00000000G | EF | 01 | FB | 00257 | CALLS | #1,PASS\$WRITELN2 | |
| | 00004140 | 8F | DF | 0025E | PUSHAF | #^F3.0 : 0441 | |
| 00000000G | EF | 01 | FB | 00264 | CALLS | #1,LIB\$WAIT : 0443 | |
| | | 00 | DD | 0026B | PUSHL | #0 | |
| | | 00 | DD | 0026D | PUSHL | #0 | |
| | | 00 | DD | 0026F | PUSHL | #0 | |
| | 00B3804B | 8F | DD | 00271 | PUSHL | #11763787 | |
| 00000000G | EF | 04 | FB | 00277 | CALLS | #4,LIB\$SIGNAL | |
| | FC | AD | 5C | D0 | 0027E | 44\$: MOVL | INDEX_LEVEL,-4(FP) : 0450 |
| | | FC | AD | 9F | 00282 | PUSHAB | -4(FPT) |
| | F8 | AD | 53 | D0 | 00285 | MOVL | ADDED_NUMBER_RECORDS,-8(FP) |
| | | F8 | AD | 9F | 00289 | PUSHAB | -8(FPT) |

| | | | | | | | |
|------|----|----|----|-------|-------|--------|-----------------------------|
| F4 | AD | | 52 | D0 | 0028C | MOVL | INIT_NUMBER_RECORDS,-12(FP) |
| | | | AD | 9F | 00290 | PUSHAB | -12(FP) |
| 0254 | CF | F4 | 03 | FB | 00293 | CALLS | #3,PROLOGUE3_BUCKETS |
| | | | 04 | 00298 | 45\$: | RET | |

: 0458

; Routine Size: 665 bytes, Routine Base: \$CODE + 00254

| | | | | | | | |
|-----------|----|-----------|-------|-----------|------------------|---------------------------------|-------------------------------------|
| | | | | 00000 | PROLOGUE3_DEPTH: | | : 0508 |
| | | | 0004 | 00000 | WORD | ^M<R2> | |
| | | | 50 | D4 | 00002 | CLRL | R0 |
| | | 51 | 50 | D0 | 00004 | 1\$: | : 0521 |
| | | | 41 | D4 | 00007 | MOVL | R0,1 |
| | | | 41 | D4 | 0000E | CLRL | INIT_NUMBER_BUCKETS[1] |
| | | | 41 | D4 | 00015 | CLRL | ADDED_NUMBER_BUCKETS[1] |
| | | | 1F | F3 | 0001C | CLRL | RECS_PER_BUCRET[1] |
| | | | 8F | C5 | 00020 | AOBLEQ | #31,R0,1\$ |
| 00000000G | E4 | 00000094G | EF | 00000200 | MULL3 | #512,1DATA+148,BYTES_PER_BUCKET | : 0534 |
| | | | EF | 00000000G | CLRL | DEEPEST | : 0539 |
| | | | 8F | DF | 00036 | PUSHAL | #0 |
| | | | 01 | FB | 0003C | CALLS | #1,CALC_BUC_OVERHEAD |
| | | | 50 | D0 | 00043 | MOVL | R0,BUCKET_OVERHEAD |
| | | | 8F | DF | 00046 | PUSHAL | #0 |
| | | | 01 | FB | 0004C | CALLS | #1,CALC_REC_OVERHEAD |
| | | | EF | D5 | 00053 | TSTL | 1DATA+228 |
| | | | 00V | 12 | 00059 | BNEQ | 3\$ |
| | | | 51 | 00000000G | EF | D0 | 0005B |
| | | | 00V | 11 | 00062 | MOVL | CUR_MAX_REC,RECORD_SIZE |
| | | | EF | D0 | 00064 | BRB | 4\$ |
| | | | EF | C1 | 0006B | 3\$: | MOVL |
| | | | 51 | D1 | 00077 | 4\$: | 4DATA+228,RECORD_SIZE |
| | | | 03 | 18 | 0007A | ADDL3 | 1DATA+204,1DATA+216,R2 |
| | | | 0000V | 31 | 0007C | CMPL | RECORD_SIZE,R2 |
| | | | 5C | C0 | 0007F | BGEQ | +3 |
| | | | 50 | C3 | 00082 | BRW | 21\$ |
| | | | 50 | D1 | 0008A | ADDL2 | BUCKET_OVERHEAD,RECORD_OVERHEAD |
| | | | 03 | 18 | 00091 | SUBL3 | RECORD_OVERHEAD,BYTES_PER_BUCKET,R0 |
| | | | 0000V | 31 | 00093 | CMPL | R0,1DATA+232 |
| | | | EF | CF | 00096 | BGEQ | +3 |
| | | | 0000V | | 0009E | BRW | 21\$ |
| | | | 0000V | | 000A0 | CASEL | 1DATA+224,#0,#2 |
| | | | 0000V | | 000A2 | .DISPL | 7\$ |
| | | | 0000V | | 000A4 | .DISPL | 8\$ |
| | | | 0000V | 31 | 000A7 | .DISPL | 12\$ |
| | | | EF | 4E | 000A7 | BRW | 16\$ |
| | | | 8F | 47 | 000AE | CVTLF | 1DATA+172,R0 |
| | | | 00V | 11 | 000BA | DIVF3 | #^F100.0,R0,RDATA+4 |
| | | | EF | 4E | 000BC | BRB | 17\$ |
| | | | 8F | 46 | 000C3 | 8\$: | CVTLF |
| | | | 00 | E1 | 000CA | DIVF2 | 1DATA+172,R0 |
| | | | 8F | 45 | 000D2 | BBC | #^F100.0,R0 |
| | | | 00V | 11 | 000DE | MULF3 | #0,BDATA+16,10\$ |
| | | | 8F | 45 | 000E0 | 10\$: | #^F0.9,R0,RDATA+4 |
| | | | 00V | 11 | 000EC | BRB | 17\$ |
| | | | EF | 4E | 000EE | 12\$: | CVTLF |
| | | | 8F | 46 | 000F5 | DIVF2 | 1DATA+172,R0 |
| | | | 00 | E1 | 000FC | BBC | #^F100.0,R0 |
| | | | 8F | 45 | 00104 | MULF3 | #0,BDATA+16,14\$ |
| | | | 00V | 11 | 00110 | BRB | 15\$ |
| | | | 8F | 45 | 00112 | 14\$: | MULF3 |

Generated Code

| | | | | | | | | | |
|--------------|----|-----------|-----|----|-------|-------|--------|--------------------------|--------|
| 00000000G | EF | 64 | BF | 9A | 0011E | 15\$: | MOVZBL | #100, IDATA | : 0601 |
| | | | 00V | 11 | 00126 | | BRB | 17\$ | |
| 00V0000000FG | EF | | 00 | E1 | 00128 | 16\$: | BBC | #0, BDATA+15, 19\$ | : 0611 |
| 00000000G | EF | 66664066 | 8F | 50 | 00130 | 17\$: | MOVF | #^F0.9, RDATA | : 0613 |
| | | | 00V | 11 | 0013B | | BRB | 20\$ | |
| 00000000G | EF | ACDA402A | 8F | 50 | 0013D | 19\$: | MOVF | #^F0.6667, RDATA | : 0617 |
| | | 00000000 | 8F | DF | 00148 | 20\$: | PUSHAL | #0 | : 0619 |
| | | 00000088G | EF | 9F | 0014E | | PUSHAB | IDATA+136 | |
| | | 000000C0G | EF | 9F | 00154 | | PUSHAB | IDATA+192 | |
| 0254 | CF | | 03 | FB | 0015A | | CALLS | #3, PROLOGUE3, BUCKETS | |
| | 50 | 00000000G | EF | D0 | 0015F | | MOVL | DEEPEST, PROLOGUE3_DEPTH | : 0624 |
| | | | 00V | 11 | 00166 | | BRB | 22\$ | |
| | | | 50 | D4 | 00168 | 21\$: | CLRL | PROLOGUE3_DEPTH | : 0630 |
| | | | | 04 | 0016A | 22\$: | RET | | : 0632 |

; Routine Size: 363 bytes, Routine Base: \$CODE + 004ED

| | | | | | | | | | |
|-----------|----------|-----------|------|------|-------|-------|--------|-------------------------------|--------|
| | | | | 001C | 00000 | | .ENTRY | NATURAL_DEPTH, ^M<R2, R3, R4> | : 0679 |
| | 5E | FE00 | CE | 9E | 00002 | | MOVAB | -512(SPT, SP | |
| | | 00000000G | EF | D4 | 00007 | | CLRL | LEAKPOINT_RIGHT | : 0732 |
| | 5C | | 01 | D0 | 0000D | | MOVL | #1, R12 | : 0738 |
| | 52 | | 5C | D0 | 00010 | 1\$: | MOVL | R12, RANGE | |
| 00000094G | EF | | 52 | D0 | 00013 | | MOVL | RANGE, IDATA+148 | : 0742 |
| 04ED | CF | | 00 | FB | 0001A | | CALLS | #0, PROLOGUE3_DEPTH | : 0743 |
| FEF8 | CD42 | | 50 | D0 | 0001F | | MOVL | R0, DEPTH-4[RANGE] | |
| | | FDFC | CD42 | D4 | 00025 | | CLRF | TALLY-4[RANGE] | : 0744 |
| E2 | 5C | | 3F | F3 | 0002A | | AOBLEQ | #63, R12, 1\$ | |
| | 5C | 00004080 | 8F | 50 | 0002E | | MOVF | #^F1.0, CURRENT_WEIGHT | : 0751 |
| | | | 50 | D4 | 00035 | | CLRL | CURRENT_DEPTH | : 0752 |
| | | | 51 | D4 | 00037 | | CLRF | CURRENT_TALLY | : 0753 |
| | 53 | | 3F | D0 | 00039 | | MOVL | #63, R3 | : 0755 |
| | 52 | | 53 | D0 | 0003C | 2\$: | MOVL | R3, RANGE | |
| | 54 | FEF8 | CD42 | DE | 0003F | | MOVAL | DEPTH-4[RANGE], R4 | : 0759 |
| | | | 64 | D5 | 00045 | | TSTL | (R4) | |
| | | | 00V | 12 | 00047 | | BNEQ | 8\$ | |
| | 3F | | 52 | D1 | 00049 | | CMPL | RANGE, #63 | : 0763 |
| | | | 00V | 18 | 0004C | | BGEQ | 7\$ | |
| | | FEFC | CD42 | D5 | 0004E | | TSTL | DEPTH[RANGE] | : 0765 |
| | | | 00V | 15 | 00053 | | BLEQ | 7\$ | |
| FE00 | CD42 | | 51 | 50 | 00055 | | MOVF | CURRENT_TALLY, TALLY[RANGE] | : 0767 |
| | | FDFC | CD42 | D4 | 0005B | 7\$: | CLRF | TALLY-4[RANGE] | : 0769 |
| | | | 00V | 11 | 00060 | | BRB | 16\$ | |
| | 50 | | 64 | D1 | 00062 | 8\$: | CMPL | (R4), CURRENT_DEPTH | : 0773 |
| | | | 00V | 15 | 00065 | | BLEQ | 12\$ | |
| | 3F | | 52 | D1 | 00067 | | CMPL | RANGE, #63 | : 0777 |
| | | | 00V | 18 | 0006A | | BGEQ | 11\$ | |
| FE00 | CD42 | | 51 | 50 | 0006C | | MOVF | CURRENT_TALLY, TALLY[RANGE] | : 0779 |
| | 50 | | 64 | D0 | 00072 | 11\$: | MOVL | (R4), CURRENT_DEPTH | : 0781 |
| | 51 | | 5C | 50 | 00075 | | MOVF | CURRENT_WEIGHT, CURRENT_TALLY | : 0782 |
| | | | 00V | 11 | 00078 | | BRB | 16\$ | |
| | 21 | | 52 | D1 | 0007A | 12\$: | CMPL | RANGE, #33 | : 0793 |
| | | | 00V | 18 | 0007D | | BGEQ | 16\$ | |
| | 51 | | 5C | 40 | 0007F | | ADDF2 | CURRENT_WEIGHT, CURRENT_TALLY | : 0795 |
| | | 00000098G | EF | D5 | 00082 | 16\$: | TSTL | IDATA+152 | : 0799 |
| | | | 00V | 12 | 00088 | | BNEQ | 18\$ | |
| 5C | CCCD3ECC | | 8F | 40 | 0008A | | ADDF2 | #^F0.1, CURRENT_WEIGHT | : 0801 |

| | | | | | | | | | |
|------|----|-------------|----|----------|-------|--------|---------------------------|---|------|
| | AB | 53 | F5 | 00091 | 18\$: | SOBGTR | R3,2\$ | | |
| | | 50 | D4 | 00094 | | CLRF | MAX_TALLY | : | 0805 |
| | | 53 | D4 | 00096 | | CLRL | MAX_RANGE | : | 0806 |
| | 51 | 01 | D0 | 00098 | | MOVL | #1,MIN_BKS | : | 0807 |
| | 5C | 01 | D0 | 0009B | | MOVL | #1,R12 | : | 0812 |
| | 52 | 5C | D0 | 0009E | 19\$: | MOVL | R12,RANGE | | |
| | 01 | FEFB CD42 | D1 | 000A1 | | CMPL | DEPTH-4[RANGE],#1 | : | 0814 |
| | | 00V | 18 | 000A7 | | BGEQ | 21\$ | | |
| 51 | 52 | 01 | C1 | 000A9 | | ADDL3 | #1,RANGE,MIN_BKS | : | 0816 |
| ED | 5C | 3F | F3 | 000AD | 21\$: | AOBLEQ | #63,R12,19\$ | | |
| | 5C | 3F | D0 | 000B1 | | MOVL | #63,R12 | : | 0822 |
| | 51 | 5C | D1 | 000B4 | | CMPL | R12,R1 | | |
| | | 00V | 19 | 000B7 | | BLSS | 25\$ | | |
| | 52 | 5C | D0 | 000B9 | 22\$: | MOVL | R12,RANGE | | |
| | 50 | FDFC CD42 | 51 | 000BC | | CMPL | TALLY-4[RANGE],MAX_TALLY | : | 0824 |
| | | 00V | 15 | 000C2 | | BLEQ | 24\$ | | |
| | 50 | FDFC CD42 | 50 | 000C4 | | MOVF | TALLY-4[RANGE],MAX_TALLY | : | 0828 |
| FFE2 | 53 | 52 | D0 | 000CA | | MOVL | RANGE,MAX_RANGE | : | 0829 |
| | 8F | 51 | F1 | 000CD | 24\$: | ACBL | R1,#-1,R12,22\$ | | |
| | 01 | 53 | D1 | 000D7 | 25\$: | CMPL | MAX_RANGE,#1 | : | 0836 |
| | | 00V | 18 | 000DA | | BGEQ | 27\$ | | |
| | 53 | 01 | D0 | 000DC | | MOVL | #1,MAX_RANGE | : | 0838 |
| | 5C | 53 | D0 | 000DF | 27\$: | MOVL | MAX_RANGE,R12 | : | 0844 |
| | 3F | 5C | D1 | 000E2 | | CMPL | R12,#63 | | |
| | | 03 | 15 | 000E5 | | BLEQ | .+3 | | |
| | | 0000V | 31 | 000E7 | | BRW | 42\$ | | |
| | 53 | 04 | C5 | 000EA | | MULL3 | #4,MAX_RANGE,R0 | | |
| 54 | 20 | 00 | EE | 000EE | | EXTV | #0,#32,DEPTH-4[R0],R4 | | |
| | 52 | 5C | D0 | 000F6 | 28\$: | MOVL | R12,RANGE | | |
| | 52 | 53 | C3 | 000F9 | | SUBL3 | MAX_RANGE,RANGE,TEMP_DIST | : | 0848 |
| | 50 | FFFFFFFFGEF | 42 | 9E 000FE | | MOVAB | COLOR_ROW-1[RANGE],R0 | : | 0850 |
| | 09 | FC | AD | D1 00106 | | CMPL | TEMP_DIST,#9 | | |
| | | 00V | 18 | 0010A | | BGEQ | 30\$ | | |
| | 60 | 03 | 90 | 0010C | | MOVAB | #3,(R0) | : | 0854 |
| | | 00V | 11 | 0010F | | BRB | 35\$ | | |
| | 08 | FC | AD | D1 00111 | 30\$: | CMPL | TEMP_DIST,#8 | : | 0858 |
| | | 00V | 15 | 00115 | | BLEQ | 33\$ | | |
| | 15 | FC | AD | D1 00117 | | CMPL | TEMP_DIST,#21 | | |
| | | 00V | 18 | 0011B | | BGEQ | 33\$ | | |
| | 60 | 02 | 90 | 0011D | | MOVAB | #2,(R0) | : | 0866 |
| | | 00V | 11 | 00120 | | BRB | 35\$ | | |
| | 60 | 01 | 90 | 00122 | 33\$: | MOVAB | #1,(R0) | : | 0874 |
| | 03 | 60 | 91 | 00125 | 35\$: | CMPL | (R0),#3 | : | 0881 |
| | | 00V | 12 | 00128 | | BNEQ | 38\$ | | |
| | 54 | FEFB CD42 | D1 | 0012A | | CMPL | DEPTH-4[RANGE],R4 | | |
| | | 00V | 13 | 00130 | | BEQL | 38\$ | | |
| | 60 | 02 | 90 | 00132 | | MOVAB | #2,(R0) | : | 0887 |
| | 54 | FEFB CD42 | D1 | 00135 | 38\$: | CMPL | DEPTH-4[RANGE],R4 | : | 0893 |
| | | 00V | 18 | 0013B | | BGEQ | 41\$ | | |
| | | 00000000G | EF | D5 0013D | | TSTL | BREAKPOINT_RIGHT | | |
| | | 00V | 12 | 00143 | | BNEQ | 41\$ | | |
| | | 0000003F | 8F | DF 00145 | | PUSHAL | #63 | : | 0899 |
| | F8 | AD | 52 | D0 0014B | | MOVL | RANGE,-8(FP) | | |
| | | F8 | AD | 9F 0014F | | PUSHAB | -8(FP) | | |
| | | 00000080G | EF | 9F 00152 | | PUSHAB | IDATA+128 | | |
| | | 00000000G | 03 | FB 00158 | | CALLS | #3,MAX_FACTOR | | |
| | | 00000000G | 5D | D0 0015F | | MOVL | R0,BREAKPOINT_RIGHT | | |

| Address | Instruction | Comment |
|-----------|---|------------------------------------|
| 00000000G | MOV R12, #63 | #63, R12, 28\$ |
| 00000000G | CMPL MAX_RANGE, #1 | MAX_RANGE, #1 |
| 00000000G | BNEQ 44\$ | 44\$ |
| 00000000G | MOV #3, COLOR_ROW | #3, COLOR_ROW |
| 00000000G | MOVL DEPTH-4[MAX_RANGE], LEFT_ADJ_RANGE | DEPTH-4[MAX_RANGE], LEFT_ADJ_RANGE |
| 00000000G | BRB 50\$ | 50\$ |
| 00000000G | MOVL DEPTH-8[MAX_RANGE], LEFT_ADJ_RANGE | DEPTH-8[MAX_RANGE], LEFT_ADJ_RANGE |
| 00000000G | SUBL3 #1, MAX_RANGE, R12 | #1, MAX_RANGE, R12 |
| 00000000G | BLEQ 50\$ | 50\$ |
| 00000000G | MOVL R12, RANGE | R12, RANGE |
| 00000000G | MOVAB COLOR_ROW-1[RANGE], R0 | COLOR_ROW-1[RANGE], R0 |
| 00000000G | CMPL DEPTH-4[RANGE], LEFT_ADJ_RANGE | DEPTH-4[RANGE], LEFT_ADJ_RANGE |
| 00000000G | BNEQ 47\$ | 47\$ |
| 00000000G | MOV #2, (R0) | #2, (R0) |
| 00000000G | BRB 48\$ | 48\$ |
| 00000000G | MOV #1, (R0) | #1, (R0) |
| 00000000G | SOBGTR R12, 45\$ | R12, 45\$ |
| 00000000G | MOVL #1, R12 | #1, R12 |
| 00000000G | MOVL R12, RANGE | R12, RANGE |
| 00000000G | CMPL DEPTH-4[RANGE], #1 | DEPTH-4[RANGE], #1 |
| 00000000G | BGEQ 53\$ | 53\$ |
| 00000000G | CLRB COLOR_ROW-1[RANGE] | COLOR_ROW-1[RANGE] |
| 00000000G | AOBLEQ #63, RT2, 51\$ | #63, RT2, 51\$ |
| 00000000G | PUSHAL #63 | #63 |
| 00000000G | MOVL MAX_RANGE, -8(FP) | MAX_RANGE, -8(FP) |
| 00000000G | PUSHAB -8(FP) | -8(FP) |
| 00000000G | PUSHAB IDATA+128 | IDATA+128 |
| 00000000G | CALLS #3, MAX_FACTOR | #3, MAX_FACTOR |
| 00000000G | MOVL R0, BREAKPOINT_MID | R0, BREAKPOINT_MID |
| 00000000G | TSTL BREAKPOINT_RIGHT | BREAKPOINT_RIGHT |
| 00000000G | BEQL .+3 | +.3 |
| 00000000G | BRW 65\$ | 65\$ |
| 00000000G | MOVL MAX_RANGE, RANGE | MAX_RANGE, RANGE |
| 00000000G | BRB 56\$ | 56\$ |
| 00000000G | INCL RANGE | RANGE |
| 00000000G | CMPL RANGE, #63 | RANGE, #63 |
| 00000000G | BGEQ 58\$ | 58\$ |
| 00000000G | CMPB COLOR_ROW-1[RANGE], #3 | COLOR_ROW-1[RANGE], #3 |
| 00000000G | BEQL 55\$ | 55\$ |
| 00000000G | TSTB COLOR_ROW-1[RANGE] | COLOR_ROW-1[RANGE] |
| 00000000G | BEQL 60\$ | 60\$ |
| 00000000G | PUSHAL #63 | #63 |
| 00000000G | MOVL RANGE, -8(FP) | RANGE, -8(FP) |
| 00000000G | PUSHAB -8(FP) | -8(FP) |
| 00000000G | PUSHAB IDATA+128 | IDATA+128 |
| 00000000G | CALLS #3, MAX_FACTOR | #3, MAX_FACTOR |
| 00000000G | MOVL R0, BREAKPOINT_RIGHT | R0, BREAKPOINT_RIGHT |
| 00000000G | BRB 65\$ | 65\$ |
| 00000000G | CMPL RANGE, MAX_RANGE | RANGE, MAX_RANGE |
| 00000000G | BEQL 62\$ | 62\$ |
| 00000000G | PUSHAL #63 | #63 |
| 00000000G | SUBL3 #1, RANGE, -8(FP) | #1, RANGE, -8(FP) |
| 00000000G | PUSHAB -8(FP) | -8(FP) |
| 00000000G | PUSHAB IDATA+128 | IDATA+128 |
| 00000000G | CALLS #3, MAX_FACTOR | #3, MAX_FACTOR |
| 00000000G | MOVL R0, BREAKPOINT_RIGHT | R0, BREAKPOINT_RIGHT |
| 00000000G | BRB 65\$ | 65\$ |

| | | | | | | | | | | | |
|----|-----------|----|-----------|------|-------|-------|-------|--------|-------------------------------|--|--------|
| | F8 | AD | 0000003F | 8F | DF | 00258 | 62\$: | PUSHAL | #63 | | : 0983 |
| | | | | 53 | D0 | 0025E | | MOVL | MAX_RANGE,-8(FP) | | |
| | | | F8 | AD | 9F | 00262 | | PUSHAB | -8(FP) | | |
| | | | 00000080G | EF | 9F | 00265 | | PUSHAB | IDATA+128 | | |
| | 00000000G | EF | | 03 | FB | 00268 | | CALLS | #3,MAX_FACTOR | | |
| 52 | 00000000G | EF | | 50 | D0 | 00272 | | MOVL | R0,BREAKPOINT_RIGHT | | |
| | | 53 | | 01 | C3 | 00279 | 65\$: | SUBL3 | #1,MAX_RANGE,RANGE | | : 0991 |
| | | | | 00V | 15 | 0027D | | BLEQ | 71\$ | | : 0993 |
| | | | | 00V | 11 | 0027F | | BRB | 68\$ | | : 0995 |
| | | | | 52 | D7 | 00281 | 67\$: | DECL | RANGE | | : 0997 |
| | 01 | | | 52 | D1 | 00283 | 68\$: | CMPL | RANGE,#1 | | |
| | | | | 00V | 15 | 00286 | | BLEQ | 71\$ | | |
| | 54 | | FEF8 | CD42 | D1 | 00288 | | CMPL | DEPTH-4[RANGE],LEFT_ADJ_RANGE | | |
| | | | | F1 | 13 | 0028E | | BEQL | 67\$ | | |
| | | | | 52 | D6 | 00290 | 71\$: | INCL | RANGE | | : 1002 |
| | 53 | | | 52 | D1 | 00292 | | CMPL | RANGE,MAX_RANGE | | : 1004 |
| | | | | 00V | 19 | 00295 | | BLSS | 73\$ | | |
| | | | 0000003F | 8F | DF | 00297 | | PUSHAL | #63 | | : 1006 |
| | F8 | AD | | 53 | D0 | 0029D | | MOVL | MAX_RANGE,-8(FP) | | |
| | | | F8 | AD | 9F | 002A1 | | PUSHAB | -8(FP) | | |
| | | | 00000080G | EF | 9F | 002A4 | | PUSHAB | IDATA+128 | | |
| | 00000000G | EF | | 03 | FB | 002AA | | CALLS | #3,MAX_FACTOR | | |
| | 00000000G | EF | | 50 | D0 | 002B1 | | MOVL | R0,BREAKPOINT_LEFT | | |
| | | | | 00V | 11 | 002B8 | | BRB | 74\$ | | |
| | | | 0000003F | 8F | DF | 002BA | 73\$: | PUSHAL | #63 | | : 1011 |
| | F8 | AD | | 52 | D0 | 002C0 | | MOVL | RANGE,-8(FP) | | |
| | | | F8 | AD | 9F | 002C4 | | PUSHAB | -8(FP) | | |
| | | | 00000080G | EF | 9F | 002C7 | | PUSHAB | IDATA+128 | | |
| | 00000000G | EF | | 03 | FB | 002CD | | CALLS | #3,MAX_FACTOR | | |
| | 00000000G | EF | | 50 | D0 | 002D4 | | MOVL | R0,BREAKPOINT_LEFT | | |
| | 50 | | 00000000G | EF | D0 | 002DB | 74\$: | MOVL | BREAKPOINT_LEFT,R0 | | : 1017 |
| | 00000000G | EF | FEF8 | CD40 | D0 | 002E2 | | MOVL | DEPTH-4[R0],DEPTHPOINT_LEFT | | |
| | 50 | | 00000000G | EF | D0 | 002EC | | MOVL | BREAKPOINT_MID,R0 | | : 1018 |
| | 00000000G | EF | FEF8 | CD40 | D0 | 002F3 | | MOVL | DEPTH-4[R0],DEPTHPOINT_MID | | |
| | 50 | | 00000000G | EF | D0 | 002FD | | MOVL | BREAKPOINT_RIGHT,R0 | | : 1019 |
| | 00000000G | EF | FEF8 | CD40 | D0 | 00304 | | MOVL | DEPTH-4[R0],DEPTHPOINT_RIGHT | | |
| | | | 00000000G | EF | 9F | 0030E | | PUSHAB | BREAKPOINT_LEFT | | : 1024 |
| | | | 00000000G | EF | 9F | 00314 | | PUSHAB | PAGEPOINT_LEFT | | |
| | | | 00000000G | EF | 9F | 0031A | | PUSHAB | NUMPOINT_LEFT | | |
| | | | 00000000G | EF | 9F | 00320 | | PUSHAB | EXAMPOINT_LEFT | | |
| | 51 | | | 5D | D0 | 00326 | | MOVL | FP,R1 | | |
| | 00V | AF | | 04 | FB | 00329 | | CALLS | #4,EXTEND_INDEX_INFO | | |
| | | | 00000000G | EF | 9F | 0032D | | PUSHAB | BREAKPOINT_MID | | : 1030 |
| | | | 00000000G | EF | 9F | 00333 | | PUSHAB | PAGEPOINT_MID | | |
| | | | 00000000G | EF | 9F | 00339 | | PUSHAB | NUMPOINT_MID | | |
| | | | 00000000G | EF | 9F | 0033F | | PUSHAB | EXAMPOINT_MID | | |
| | 51 | | | 5D | D0 | 00345 | | MOVL | FP,R1 | | |
| | 00V | AF | | 04 | FB | 00348 | | CALLS | #4,EXTEND_INDEX_INFO | | |
| | | | 00000000G | EF | 9F | 0034C | | PUSHAB | BREAKPOINT_RIGHT | | : 1036 |
| | | | 00000000G | EF | 9F | 00352 | | PUSHAB | PAGEPOINT_RIGHT | | |
| | | | 00000000G | EF | 9F | 00358 | | PUSHAB | NUMPOINT_RIGHT | | |
| | | | 00000000G | EF | 9F | 0035E | | PUSHAB | EXAMPOINT_RIGHT | | |
| | 51 | | | 5D | D0 | 00364 | | MOVL | FP,R1 | | |
| | 00V | AF | | 04 | FB | 00367 | | CALLS | #4,EXTEND_INDEX_INFO | | |
| | 50 | | 00000000G | EF | D0 | 0036B | | MOVL | BREAKPOINT_MID,NATURAL_DEPTH | | : 1042 |
| | | | | 04 | 00372 | | | RET | | | : 1044 |

; Routine Size: 883 bytes, Routine Base: \$CODE + 00658

| PC | OP | OP2 | OP3 | OP4 | OP5 | OP6 | OP7 | OP8 | OP9 | OP10 | OP11 | OP12 | OP13 | OP14 | OP15 | OP16 | OP17 | OP18 | OP19 | OP20 | OP21 | OP22 | OP23 | OP24 | OP25 | OP26 | OP27 | OP28 | OP29 | OP30 | OP31 | OP32 | OP33 | OP34 | OP35 | OP36 | OP37 | OP38 | OP39 | OP40 | OP41 | OP42 | OP43 | OP44 | OP45 | OP46 | OP47 | OP48 | OP49 | OP50 | OP51 | OP52 | OP53 | OP54 | OP55 | OP56 | OP57 | OP58 | OP59 | OP60 | OP61 | OP62 | OP63 | OP64 | OP65 | OP66 | OP67 | OP68 | OP69 | OP70 | OP71 | OP72 | OP73 | OP74 | OP75 | OP76 | OP77 | OP78 | OP79 | OP80 | OP81 | OP82 | OP83 | OP84 | OP85 | OP86 | OP87 | OP88 | OP89 | OP90 | OP91 | OP92 | OP93 | OP94 | OP95 | OP96 | OP97 | OP98 | OP99 | OP100 | OP101 | OP102 | OP103 | OP104 | OP105 | OP106 | OP107 | OP108 | OP109 | OP110 | OP111 | OP112 | OP113 | OP114 | OP115 | OP116 | OP117 | OP118 | OP119 | OP120 | OP121 | OP122 | OP123 | OP124 | OP125 | OP126 | OP127 | OP128 | OP129 | OP130 | OP131 | OP132 | OP133 | OP134 | OP135 | OP136 | OP137 | OP138 | OP139 | OP140 | OP141 | OP142 | OP143 | OP144 | OP145 | OP146 | OP147 | OP148 | OP149 | OP150 | OP151 | OP152 | OP153 | OP154 | OP155 | OP156 | OP157 | OP158 | OP159 | OP160 | OP161 | OP162 | OP163 | OP164 | OP165 | OP166 | OP167 | OP168 | OP169 | OP170 | OP171 | OP172 | OP173 | OP174 | OP175 | OP176 | OP177 | OP178 | OP179 | OP180 | OP181 | OP182 | OP183 | OP184 | OP185 | OP186 | OP187 | OP188 | OP189 | OP190 | OP191 | OP192 | OP193 | OP194 | OP195 | OP196 | OP197 | OP198 | OP199 | OP200 | OP201 | OP202 | OP203 | OP204 | OP205 | OP206 | OP207 | OP208 | OP209 | OP210 | OP211 | OP212 | OP213 | OP214 | OP215 | OP216 | OP217 | OP218 | OP219 | OP220 | OP221 | OP222 | OP223 | OP224 | OP225 | OP226 | OP227 | OP228 | OP229 | OP230 | OP231 | OP232 | OP233 | OP234 | OP235 | OP236 | OP237 | OP238 | OP239 | OP240 | OP241 | OP242 | OP243 | OP244 | OP245 | OP246 | OP247 | OP248 | OP249 | OP250 | OP251 | OP252 | OP253 | OP254 | OP255 | OP256 | OP257 | OP258 | OP259 | OP260 | OP261 | OP262 | OP263 | OP264 | OP265 | OP266 | OP267 | OP268 | OP269 | OP270 | OP271 | OP272 | OP273 | OP274 | OP275 | OP276 | OP277 | OP278 | OP279 | OP280 | OP281 | OP282 | OP283 | OP284 | OP285 | OP286 | OP287 | OP288 | OP289 | OP290 | OP291 | OP292 | OP293 | OP294 | OP295 | OP296 | OP297 | OP298 | OP299 | OP300 | OP301 | OP302 | OP303 | OP304 | OP305 | OP306 | OP307 | OP308 | OP309 | OP310 | OP311 | OP312 | OP313 | OP314 | OP315 | OP316 | OP317 | OP318 | OP319 | OP320 | OP321 | OP322 | OP323 | OP324 | OP325 | OP326 | OP327 | OP328 | OP329 | OP330 | OP331 | OP332 | OP333 | OP334 | OP335 | OP336 | OP337 | OP338 | OP339 | OP340 | OP341 | OP342 | OP343 | OP344 | OP345 | OP346 | OP347 | OP348 | OP349 | OP350 | OP351 | OP352 | OP353 | OP354 | OP355 | OP356 | OP357 | OP358 | OP359 | OP360 | OP361 | OP362 | OP363 | OP364 | OP365 | OP366 | OP367 | OP368 | OP369 | OP370 | OP371 | OP372 | OP373 | OP374 | OP375 | OP376 | OP377 | OP378 | OP379 | OP380 | OP381 | OP382 | OP383 | OP384 | OP385 | OP386 | OP387 | OP388 | OP389 | OP390 | OP391 | OP392 | OP393 | OP394 | OP395 | OP396 | OP397 | OP398 | OP399 | OP400 | OP401 | OP402 | OP403 | OP404 | OP405 | OP406 | OP407 | OP408 | OP409 | OP410 | OP411 | OP412 | OP413 | OP414 | OP415 | OP416 | OP417 | OP418 | OP419 |
|----|----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
|----|----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|

; Routine Size: 79 bytes, Routine Base: \$CODE + 009CB

| Address | Instruction | Comment | Address | Instruction | Comment |
|-----------|--------------|---------|-------------|-------------|---------------------------------------|
| 00000000G | EF FFFFF601 | | 007C 00000 | PLOT_GRAPH: | |
| | 00000000G | | 00000 00000 | .WORD | *M<R2,R3,R4,R5,R6> |
| | 00000000G | | 00000 00002 | MOVAB | -100(SP), SP |
| | 00V00000000G | | 00000 00006 | CMPL | IDATA+280, #5 |
| | | | 00000 0000D | BNEQ | 2\$ |
| | | | 00000 0000F | CLRL | GRAPH_TYPE |
| | | | 00000 00015 | MOVC3 | #32, C.AAB, Y LABEL |
| | | | 00000 00021 | MOVL | CURRENT_GRAPH_INDEX, TEMP_INTEGER |
| | | | 00000 00028 | MOVL | LAST_GRAPH_INDEX, CURRENT_GRAPH_INDEX |
| | | | 00000 00033 | MOVL | TEMP_INTEGER, LAST_GRAPH_INDEX |
| | | | 00000 0003A | BBC | #0, FIRST_PLOT, 4\$ |
| | | | 00000 00042 | MCOML | #0, GRAPH_SWITCH |
| | | | 00000 00045 | CLRL | CURRENT_GRAPH_INDEX |
| | | | 00000 0004B | MOVL | #1, LAST_GRAPH_INDEX |
| | | | 00000 00052 | BRB | 8\$ |
| | | | 00000 00054 | CMPL | IDATA+280, #5 |
| | | | 00000 0005B | BNEQ | 6\$ |
| | | | 00000 0005D | MOVL | LAST_GRAPH_INDEX, GRAPH_SWITCH |
| | | | 00000 00064 | BRB | 8\$ |
| | | | 00000 00066 | MOVL | #1, GRAPH_SWITCH |
| | | | 00000 00069 | CMPL | IDATA+280, #5 |
| | | | 00000 00070 | BEQL | 10\$ |
| | | | 00000 00072 | CLRL | CURRENT_GRAPH_INDEX |
| | | | 00000 00078 | BRB | 13\$ |
| | | | 00000 0007A | CLRL | R2 |
| | | | 00000 0007C | MOVL | R2, RANGE |
| | | | 00000 0007F | ADDL3 | #1, RANGE, IDATA+148 |
| | | | 00000 00087 | MULL3 | #32, CURRENT_GRAPH_INDEX, R4 |
| | | | 00000 0008F | ADDL2 | RANGE, R4 |
| | | | 00000 00092 | CALLS | #0, PROLOGUE3 DEPTH |
| | | | 00000 00097 | MOVL | R0, XY_PLOT[R4] |
| | | | 00000 0009F | AOBLEQ | #31, R2, 11\$ |
| | | | 00000 000A3 | CALLS | #0, NATURAL DEPTH |
| | | | 00000 000AB | MOVL | R0, TEMP_INTEGER |
| | | | 00000 000AB | CLRL | R12 |

Generated Code

```
52 00000000G EF 50 5C D0 000AD 12$: MOVL R12,TEMP_INT2
52 00000000G EF 50 C5 000B0 MULL3 #32,CURRENT_GRAPH_INDEX,R2 ; 1183
00000000GEF 42 00000000GEF 40 9A 000B8 ADDL2 TEMP_INT2,R2
E1 00000000G EF 5C 1F F3 000C8 MOVZBL COLOR_ROW[TEMP_INT2],COLOR_PLOT[R2]
03 00000000G EF 00 00 E1 000CC 13$: AOBLEQ #31,RT2,12$ ; 1187
00v00000000G EF 00 00 31 000D4 BRW 18$
FF FFFF55D EF 9F 000DF BBC #0,REGIS,17$ ; 1194
00000000G EF 05 DD 000E5 PUSHAB C,AAC ; 1202
00000000G EF 03 9F 000E7 PUSHAB PASS$FV_OUTPUT
00000000G EF 01 9F 000ED CALLS #3,PASS$WRITE_STRING
00000000G EF 01 FB 000FA CALLS #1,PASS$WRITELN2
00000000G EF 9F 00101 PUSHAB COL_ONE ; 1207
00000000G EF 9F 00107 PUSHAB LINE_ONE
D4 AD FFFF52D EF 02 FB 0010D CALLS #2,LIB$ERASE_PAGE ; 1214
D8 AD 00000000G EF 2C 28 00114 17$: MOVC3 #44,C.AAD,-44(FP)
E4 AD 00000000G EF 9E 0011D MOVAB COLOR_PLOT,-40(FP)
D4 AD 9F 0012D MOVAB COLOR_PLOT,-28(FP)
CC AD 010E0020 8F D0 00130 PUSHAB -44(FP)
D0 AD 00000000G EF 9E 00138 MOVL #17694752,-52(FP)
CC AD 9F 00140 MOVAB Y_LABEL,-48(FP)
00000018G EF 9F 00143 PUSHAB -52(FP)
00000014G EF 9F 00149 PUSHAB IDATA+24
00000010G EF 9F 0014F PUSHAB IDATA+20
C8 AD 56 D0 00155 PUSHAB IDATA+16
C8 AD 9F 00159 MOVL GRAPH_SWITCH,-56(FP)
00000000G EF 9F 0015C PUSHAB -56(FP)
9C AD FFFF50B EF 2C 28 00162 PUSHAB CURRENT_GRAPH_INDEX
AO AD 00000000G EF 9E 0016B MOVC3 #44,C.AXE,-100(FP)
AC AD 00000000G EF 9E 00173 MOVAB XY_PLOT,-96(FP)
9C AD 9F 0017B MOVAB XY_PLOT,-84(FP)
00000000G EF 9F 0017E PUSHAB -100(FP)
00v00000000G EF 09 FB 00184 PUSHAB GRAPH_TYPE
00000000G EF 00 00 E1 0018B 18$: CALLS #9,EDF$GRAPH ; 1232
00000000G EF 94 00193 BBC #0,DEC CRT,20$ ; 1234
04 00199 20$: CLRB FIRST_PLOT ; 1236
RET
```

; Routine Size: 410 bytes, Routine Base: \$CODE + 00A1A

```
0000 00000 WARN_OF_ERASE: ; 1283
03 00000000G EF 00 0000 .WORD ^M<>
00000000G EF 0000V 31 00002 BBC #0,AUTO_TUNE,..+3 ; 1287
00000000G EF 00V D1 0000A BRW 11$
50 00000000G EF 00V 12 00018 CMPL DEF_HEAD,DEF_TAIL ; 1295
09 19 A0 91 00021 BNEQ 3$
03 12 00025 MOVL DEF_HEAD,R0
0000V 31 00027 CMPB 25(R0),#9
06 00000108G EF D1 0002A 3$: BNEQ +3 ; 1303
00V 13 00031 BRW 11$
05 00000108G EF D1 00033 CMPL IDATA+264,#6
03 13 0003A BEQL 5$
0000V 31 0003C BEQL IDATA+264,#5
BRW 7$
```



```
03 00000000G EF 00 00 0003F 5$: BBS #0,ISAM_ORG,..+3
00000000G EF 0000V 31 00047 7$
00000000G EF 9F 0004A PUSHAB SHIFT ; 1311
00000000G EF 04 DD 00050 PUSHAB #4
00000000G EF 9F 00052 PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 00058 CALLS #3,PASSWRITE_STRING
00000000G EF 9F 0005F PUSHAB ANSI_REVERSE
00000000G EF 04 DD 00065 PUSHAB #4
00000000G EF 9F 00067 PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 0006D CALLS #3,PASSWRITE_STRING
00000000G EF FFFF48E EF 9F 00074 PUSHAB C.AAF
00000000G EF 15 DD 0007A PUSHAB #21
00000000G EF 9F 0007C PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 00082 CALLS #3,PASSWRITE_STRING
00000000G EF 03 DD 00089 PUSHAB #3
00000000G EF 0000084G EF DD 0008B PUSHAB IDATA+132
00000000G EF 00000000G EF 9F 00091 PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 00097 CALLS #3,PASSWRITE_INTEGER
00000000G EF FFFF47C EF 9F 0009E PUSHAB C.AAG
00000000G EF 12 DD 000A4 PUSHAB #18
00000000G EF 9F 000A6 PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 000AC CALLS #3,PASSWRITE_STRING
00000000G EF 9F 000B3 PUSHAB CRLF
00000000G EF 02 DD 000B9 PUSHAB #2
00000000G EF 9F 000BB PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 000C1 CALLS #3,PASSWRITE_STRING
00000000G EF 00V 11 000C8 BRB 8$
00000000G EF 9F 000CA 7$: PUSHAB SHIFT ; 1317
00000000G EF 04 DD 000D0 PUSHAB #4
00000000G EF 9F 000D2 PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 000D8 CALLS #3,PASSWRITE_STRING
00000000G EF 9F 000DF PUSHAB ANSI_REVERSE
00000000G EF 04 DD 000E5 PUSHAB #4
00000000G EF 9F 000E7 PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 000ED CALLS #3,PASSWRITE_STRING
00000000G EF FFFF43A EF 9F 000F4 PUSHAB C.AAH
00000000G EF 2A DD 000FA PUSHAB #42
00000000G EF 9F 000FC PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 00102 CALLS #3,PASSWRITE_STRING
00000000G EF 9F 00109 PUSHAB ANSI_RESET
00000000G EF 04 DD 0010F PUSHAB #4
00000000G EF 9F 00111 PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 00117 CALLS #3,PASSWRITE_STRING
00000000G EF 9F 0011E PUSHAB CRLF
00000000G EF 02 DD 00124 PUSHAB #2
00000000G EF 9F 00126 PUSHAB PASSFV OUTPUT
00000000G EF 03 FB 0012C CALLS #3,PASSWRITE_STRING
00000000G EF 9F 00133 PUSHAB PASSFV OUTPUT
00000000G EF 01 FB 00139 CALLS #1,PASSWRITELN2
00000000G EF 0000001F 8F DF 00140 8$: PUSHAB #31 ; 1321
00000000G EF 01 FB 00146 11$: CALLS #1,QUERY ; 1327
00000000G EF 04 0014D RET
```

; Routine Size: 334 bytes, Routine Base: \$CODE + 00BB4

0000 00000 NON_KEY_DEF:
0000 00000 .WORD *M<>

; 1374

| Generated Code | | | | | | | | |
|----------------|----|-----------|-----|----|-------|--------|---------------------|--------|
| | | 0000000C | 8F | DF | 00002 | PUSHAL | #12 | : 1381 |
| 00000000G | EF | | 01 | FB | 00008 | CALLS | #1, QUERY | |
| | | 0000000B | 8F | DF | 0000F | PUSHAL | #11 | : 1382 |
| 00000000G | EF | | 01 | FB | 00015 | CALLS | #1, QUERY | |
| | | 00000027 | 8F | DF | 0001C | PUSHAL | #39 | : 1383 |
| 00000000G | EF | | 01 | FB | 00022 | CALLS | #1, QUERY | |
| 00V00000005G | EF | | 00 | E1 | 00029 | BBC | #0, BDATA+5, 7\$ | : 1388 |
| 00000000G | EF | | 00 | FB | 00031 | CALLS | #0, MAKE_SCRATCH | : 1392 |
| | 5C | 00000000G | EF | DO | 00038 | MOVL | DEF_SCRATCH, R12 | : 1394 |
| | | 11 | AC | 9F | 0003F | PUSHAB | 17(R12) | : 1401 |
| | | 00000010G | EF | 9F | 00042 | PUSHAB | SDATA+16 | |
| 00000000G | EF | | 02 | FB | 00048 | CALLS | #2, LIB\$SCOPY_DXDX | |
| | | 00000010G | EF | 9F | 0004F | PUSHAB | SDATA+16 | : 1402 |
| 00000000G | EF | | 01 | FB | 00055 | CALLS | #1, STR\$FREE1_DX | |
| 19 | AC | | 0F | 90 | 0005C | MOVB | #15, 25(R12) | : 1404 |
| | | | 6C | 94 | 00060 | CLRB | (R12) | : 1405 |
| | | 00000000 | 8F | DF | 00062 | PUSHAL | #0 | : 1407 |
| 00000000G | EF | | 01 | FB | 00068 | CALLS | #1, INSERT_IN_ORDER | |
| | | 00000000 | 00V | 11 | 0006F | BRB | 10\$ | |
| | | 00000000 | 8F | DF | 00071 | PUSHAL | #0 | : 1417 |
| | | 00 | 8F | 9F | 00077 | PUSHAB | #0 | |
| | | 00000000 | 8F | DF | 0007A | PUSHAL | #0 | |
| | | 0F | 8F | 9F | 00080 | PUSHAB | #15 | |
| | | 00 | 8F | 9F | 00083 | PUSHAB | #0 | |
| 00000000G | EF | | 05 | FB | 00086 | CALLS | #5, FIND_OBJECT | |
| 00V | | | 50 | E9 | 0008D | BLBC | R0, 10\$ | |
| 00000000G | EF | | 00 | FB | 00090 | CALLS | #0, DELETE_CURRENT | : 1419 |
| 00000000G | EF | | 00 | FB | 00097 | CALLS | #0, MAKE_SCRATCH | : 1423 |
| | 50 | 00000000G | EF | DO | 0009E | MOVL | DEF_SCRATCH, R0 | : 1425 |
| | | | 60 | 94 | 000A5 | CLRB | (R0) | : 1432 |
| 19 | AO | | 0E | 90 | 000A7 | MOVB | #14, 25(R0) | : 1433 |
| | | 00000000 | 8F | DF | 000AB | PUSHAL | #0 | : 1435 |
| 00000000G | EF | | 01 | FB | 000B1 | CALLS | #1, INSERT_IN_ORDER | |
| 00000000G | EF | | 00 | FB | 000B8 | CALLS | #0, MAKE_SCRATCH | : 1439 |
| | 50 | 00000000G | EF | DO | 000BF | MOVL | DEF_SCRATCH, R0 | : 1441 |
| 19 | AO | | 0E | 90 | 000C6 | MOVB | #14, 25(R0) | : 1448 |
| 1E | AO | 95 | 8F | 90 | 000CA | MOVB | #-107, 30(R0) | : 1449 |
| 23 | AO | | 1C | DO | 000CF | MOVL | #28, 35(R0) | : 1450 |
| | | 00000000 | 8F | DF | 000D3 | PUSHAL | #0 | : 1452 |
| 00000000G | EF | | 01 | FB | 000D9 | CALLS | #1, INSERT_IN_ORDER | |
| 00000000G | EF | | 00 | FB | 000E0 | CALLS | #0, MAKE_SCRATCH | : 1456 |
| | 50 | 00000000G | EF | DO | 000E7 | MOVL | DEF_SCRATCH, R0 | : 1458 |
| | | | 60 | 94 | 000EE | CLRB | (R0) | : 1465 |
| 19 | AO | | 08 | 90 | 000F0 | MOVB | #8, 25(R0) | : 1466 |
| | | 00000000 | 8F | DF | 000F4 | PUSHAL | #0 | : 1468 |
| 00000000G | EF | | 01 | FB | 000FA | CALLS | #1, INSERT_IN_ORDER | |
| 00V00000004G | EF | | 00 | E1 | 00101 | BBC | #0, BDATA+4, 17\$ | : 1475 |
| 00000000G | EF | | 00 | FB | 00109 | CALLS | #0, MAKE_SCRATCH | : 1479 |
| | 5C | 00000000G | EF | DO | 00110 | MOVL | DEF_SCRATCH, R12 | : 1481 |
| | | 11 | AC | 9F | 00117 | PUSHAB | 17(R12) | : 1485 |
| | | 00000008G | EF | 9F | 0011A | PUSHAB | SDATA+8 | |
| 00000000G | EF | | 02 | FB | 00120 | CALLS | #2, LIB\$SCOPY_DXDX | |
| | | 00000008G | EF | 9F | 00127 | PUSHAB | SDATA+8 | : 1486 |
| 00000000G | EF | | 01 | FB | 0012D | CALLS | #1, STR\$FREE1_DX | |
| 19 | AC | | 08 | 90 | 00134 | MOVB | #8, 25(R12) | : 1488 |
| 1E | AC | 5E | 8F | 90 | 00138 | MOVB | #94, 30(R12) | : 1489 |
| | | 00000000 | 8F | DF | 0013D | PUSHAL | #0 | : 1491 |

| Generated Code | | | |
|----------------|--------------|-------|---------------------------------|
| 00000000G | EF | 01 | FB 00143 |
| | | 00V | 11 0014A |
| | 00000000 | 8F | DF 0014C 17\$: |
| | 5E | 8F | 9F 00152 |
| | 00000000 | 8F | DF 00155 |
| | 08 | 8F | 9F 0015B |
| | 01 | 8F | 9F 0015E |
| 00000000G | EF | 05 | FB 00161 |
| | 00V | 50 | E9 00168 |
| 00000000G | EF | 00 | FB 0016B |
| 00000000G | EF | 00 | FB 00172 20\$: |
| | 50 00000000G | EF | DF 00179 |
| 19 | A0 | 08 | 90 00180 |
| 1E | A0 | 8F | 90 00184 |
| 04 | 02 00000108G | EF | CF 00189 |
| | | 0000V | 00191 |
| | | 0000V | 00193 |
| | | 0000V | 00195 |
| | | 0000V | 00197 |
| | | 0000V | 00199 |
| | | 00V | 11 0019B |
| 23 | A0 | 1F | DF 0019D 22\$: |
| | | 00V | 11 001A1 |
| 23 | A0 | 1D | DF 001A3 23\$: |
| | | 00V | 11 001A7 |
| 23 | A0 | 1E | DF 001A9 24\$: |
| | | 00V | 11 001AD |
| | | | 001AF 25\$: |
| | 00000000 | 8F | DF 001AF 26\$: |
| 00000000G | EF | 01 | FB 001B5 |
| 00000000G | EF | 00 | FB 001BC |
| | 50 00000000G | EF | DF 001C3 |
| | | 60 | 94 001CA |
| 19 | A0 | 0C | 90 001CC |
| | 00000000 | 8F | DF 001D0 |
| 00000000G | EF | 01 | FB 001D6 |
| | 04 00000108G | EF | D1 001DD |
| | | 00V | 12 001E4 |
| 00000000G | EF | 00 | FB 001E6 |
| | 50 00000000G | EF | DF 001ED |
| | | 0C | 90 001F4 |
| 19 | A0 | 8F | 90 001F8 |
| 1E | A0 | 8F | 90 001FD |
| 28 | A0 00000011G | EF | 90 001FD |
| | 00000000 | 8F | DF 00205 |
| 00000000G | EF | 01 | FB 0020B |
| 00000000G | EF | 00 | FB 00212 30\$: |
| | 50 00000000G | EF | DF 00219 |
| | | 0C | 90 00220 |
| 19 | A0 | 8F | 90 00224 |
| 1E | A0 | 8F | 90 00229 |
| 23 | A0 0000009CG | EF | DF 00231 |
| | 00000000 | 8F | DF 00231 |
| 00000000G | EF | 01 | FB 00237 |
| | 04 00000108G | EF | D1 0023E |
| | | 00V | 13 00245 |
| | 03 00000108G | EF | D1 00247 |
| | | 00V | 12 0024E |
| | 0F 00000100G | EF | D1 00250 33\$: |
| | | | CALLS #1, INSERT_IN_ORDER |
| | | | BRB 20\$ |
| | | | PUSHAL #0 ; 1501 |
| | | | PUSHAB #94 |
| | | | PUSHAL #0 |
| | | | PUSHAB #8 |
| | | | PUSHAB #1 |
| | | | CALLS #5, FIND_OBJECT |
| | | | BLBC R0, 20\$ |
| | | | CALLS #0, DELETE_CURRENT ; 1503 |
| | | | CALLS #0, MAKE_SCRATCH ; 1507 |
| | | | MOVL DEF_SCRATCH, R0 ; 1509 |
| | | | MOVB #8, 25(R0) ; 1516 |
| | | | MOVB #98, 30(R0) ; 1517 |
| | | | CASEL IDATA+264, #2, #4 ; 1519 |
| | | | .DISPL 22\$ |
| | | | .DISPL 24\$ |
| | | | .DISPL 23\$ |
| | | | .DISPL 22\$ |
| | | | .DISPL 22\$ |
| | | | .DISPL 22\$ |
| | | | BRB 25\$ |
| | | | MOVL #31, 35(R0) ; 1523 |
| | | | BRB 26\$ |
| | | | MOVL #29, 35(R0) ; 1524 |
| | | | BRB 26\$ |
| | | | MOVL #30, 35(R0) ; 1525 |
| | | | BRB 26\$ |
| | | | PUSHAL #0 ; 1533 |
| | | | CALLS #1, INSERT_IN_ORDER |
| | | | CALLS #0, MAKE_SCRATCH ; 1537 |
| | | | MOVL DEF_SCRATCH, R0 ; 1539 |
| | | | CLRB (R0) ; 1546 |
| | | | MOVB #12, 25(R0) ; 1547 |
| | | | PUSHAL #0 ; 1549 |
| | | | CALLS #1, INSERT_IN_ORDER |
| | | | CMPL IDATA+264, #4 ; 1553 |
| | | | BNEQ 30\$ |
| | | | CALLS #0, MAKE_SCRATCH ; 1560 |
| | | | MOVL DEF_SCRATCH, R0 ; 1562 |
| | | | MOVB #12, 25(R0) ; 1566 |
| | | | MOVB #-120, 30(R0) ; 1567 |
| | | | MOVB BDATA+17, 43(R0) ; 1568 |
| | | | PUSHAL #0 ; 1570 |
| | | | CALLS #1, INSERT_IN_ORDER |
| | | | CALLS #0, MAKE_SCRATCH ; 1579 |
| | | | MOVL DEF_SCRATCH, R0 ; 1581 |
| | | | MOVB #12, 25(R0) ; 1585 |
| | | | MOVB #-119, 30(R0) ; 1586 |
| | | | MOVL IDATA+156, 35(R0) ; 1587 |
| | | | PUSHAL #0 ; 1589 |
| | | | CALLS #1, INSERT_IN_ORDER |
| | | | CMPL IDATA+264, #4 ; 1593 |
| | | | BEQL 33\$ |
| | | | CMPL IDATA+264, #3 |
| | | | BNEQ 36\$ |
| | | | CMPL IDATA+256, #15 |

Generated Code

| | | | | | | | | | |
|-----------|----|-----------|-----|-------|-------|--------|--------------------|---|------|
| 00000000G | EF | | 00V | 12 | 00257 | BNEQ | 36\$ | | |
| | 50 | 00000000G | 00 | FB | 00259 | CALLS | #0,MAKE_SCRATCH | : | 1606 |
| 19 | A0 | | EF | D0 | 00260 | MOVL | DEF_SCRATCH,R0 | : | 1608 |
| 1E | A0 | 8A | 0C | 90 | 00267 | MOVB | #12,25(R0) | : | 1612 |
| 27 | A0 | 000000A0G | 8F | 90 | 0026B | MOVB | #-118,30(R0) | : | 1613 |
| | | 00000000 | EF | D0 | 00270 | MOVL | IDATA+160,39(R0) | : | 1614 |
| | | | 8F | DF | 00278 | PUSHAL | #0 | : | 1616 |
| 00000000G | EF | | 01 | FB | 0027E | CALLS | #1,INSERT_IN_ORDER | | |
| 00000000G | EF | | 00 | FB | 00285 | CALLS | #0,MAKE_SCRATCH | : | 1622 |
| | 50 | 00000000G | EF | D0 | 0028C | MOVL | DEF_SCRATCH,R0 | : | 1624 |
| 19 | A0 | | 0C | 90 | 00293 | MOVB | #12,25(R0) | : | 1631 |
| 1E | A0 | 8B | 8F | 90 | 00297 | MOVB | #-117,30(R0) | : | 1632 |
| 23 | A0 | 00000100G | EF | D0 | 0029C | MOVL | IDATA+256,35(R0) | : | 1633 |
| | | 00000000 | 8F | DF | 002A4 | PUSHAL | #0 | : | 1635 |
| 00000000G | EF | | 01 | FB | 002AA | CALLS | #1,INSERT_IN_ORDER | | |
| 00000000G | EF | | 00 | FB | 002B1 | CALLS | #0,MAKE_SCRATCH | : | 1642 |
| | 50 | 00000000G | EF | D0 | 002B8 | MOVL | DEF_SCRATCH,R0 | : | 1644 |
| 19 | A0 | | 0C | 90 | 002BF | MOVB | #12,25(R0) | : | 1648 |
| 1E | A0 | 8C | 8F | 90 | 002C3 | MOVB | #-116,30(R0) | : | 1649 |
| 27 | A0 | 000000E4G | EF | D0 | 002C8 | MOVL | IDATA+228,39(R0) | : | 1650 |
| | | 00000000 | 8F | DF | 002D0 | PUSHAL | #0 | : | 1652 |
| 00000000G | EF | | 01 | FB | 002D6 | CALLS | #1,INSERT_IN_ORDER | | |
| | | | 04 | 002DD | RET | | | : | 1656 |

; Routine Size: 734 bytes, Routine Base: \$CODE + 00D02

| | | | | | | | | | |
|--------------|----|-----------|-----|-------|-------|---------------|----------------------------|---|------|
| | | | | 0000 | 00000 | CALC_ALLOC: | | : | 1701 |
| | 5C | 04 | BC | D0 | 00002 | WORD | ^M<> | | |
| | 5C | | 5C | 4E | 00006 | MOVL | #4(R12),RECORD_TOT | | |
| | 50 | 000000C0G | EF | 4E | 00009 | CVTLF | RECORD_TOT,BYTES_REAL | : | 1714 |
| | 08 | | 50 | 51 | 00010 | CVTLF | IDATA+T92,NUMRECS_REAL | : | 1715 |
| | | | 00V | 18 | 00013 | CMPL | NUMRECS_REAL,#^F1.0 | : | 1717 |
| | 50 | 00004080 | 8F | 50 | 00015 | BGEQ | 2\$ | | |
| | 5C | 00004500 | 8F | 46 | 0001C | MOVF | #^F1.0,NUMRECS_REAL | : | 1719 |
| 51 6B284F6E. | 8F | | 50 | 47 | 00023 | DIVF2 | #^F512.0,BYTES_REAL | : | 1721 |
| | 51 | | 5C | 51 | 0002B | DIVF3 | NUMRECS_REAL,#^F1.0E+09,R1 | : | 1723 |
| | | | 00V | 15 | 0002E | CMPL | RATIO,RT | | |
| | 51 | 3B9AC9FF | 8F | D0 | 00030 | BLEQ | 4\$ | | |
| | | | 00V | 11 | 00037 | MOVL | #999999999,CALC_ALLOC | : | 1725 |
| | 50 | | 5C | 44 | 00039 | BRB | 5\$ | | |
| | 51 | | 50 | 4B | 0003C | MULF2 | RATIO,NUMRECS_REAL | : | 1729 |
| | 50 | | 51 | D0 | 0003F | CVTRFL | NUMRECS_REAL,CALC_ALLOC | : | 1731 |
| | | | 04 | 00042 | MOVL | CALC_ALLOC,R0 | | | |
| | | | | | RET | | | | |

; Routine Size: 67 bytes, Routine Base: \$CODE + 00FE0

| | | | | | | | | | | |
|---------------|----|-----------|----|------|-------|----------|----------------------------|------|------|------|
| | | | | 0000 | 00000 | SEQ_DEF: | WORD | ^M<> | : | 1778 |
| | 5E | | 04 | C2 | 00002 | SUBL2 | #4,SP | | | |
| | 50 | 000000E8G | EF | D0 | 00005 | MOVL | IDATA+232,RECORD_TOT | : | 1791 | |
| 00V000000000G | EF | | 00 | E1 | 0000C | BBC | #0,VARIABLE_RECORDS,2\$ | : | 1793 | |
| | 50 | | 02 | C0 | 00014 | ADDL2 | #2,RECORD_TOT | : | 1795 | |
| 00V00000011G | EF | | 00 | E0 | 00017 | BBS | #0,BDATA+T7,4\$ | : | 1800 | |
| | 51 | | 50 | 4E | 0001F | CVTLF | RECORD_TOT,R1 | : | 1808 | |
| 51 00004500 | 8F | | 51 | 47 | 00022 | DIVF3 | R1,#^F512.0,RECORD_REAL | | | |
| | 51 | | 51 | 4A | 0002A | CVTLF | RECORD_REAL,RECORD_INT | : | 1809 | |
| 50 00000200 | 8F | | 51 | C7 | 0002D | DIVL3 | RECORD_INT,#512,RECORD_TOT | : | 1810 | |

Generated Code

| | | | | | | | | |
|-----------|----|-----------|----|-------|-------|--------|--------------------|--------|
| FC | AD | 50 | DO | 00035 | 4\$: | MOVL | RECORD_TOT,-4(FP) | : 1814 |
| | | AD | 9F | 00039 | | PUSHAB | -4(FP) | |
| 0FE0 | CF | FC | 01 | FB | 0003C | CALLS | #1,CALC_ALLOC | |
| | 5C | | 50 | DO | 00041 | MOVL | R0,ALLOC | |
| 00000000G | EF | | 00 | FB | 00044 | CALLS | #0,MAKE_SCRATCH | : 1819 |
| | 50 | 00000000G | EF | DO | 0004B | MOVL | DEF_SCRATCH,R0 | : 1821 |
| 19 | AO | | 08 | 90 | 00052 | MOVB | #8,25(R0) | : 1828 |
| 1E | AO | 48 | 8F | 90 | 00056 | MOVB | #72,30(R0) | : 1829 |
| 27 | AO | | 5C | DO | 0005B | MOVL | ALLOC,39(R0) | : 1830 |
| | | 00000000 | 8F | DF | 0005F | PUSHAL | #0 | : 1832 |
| 00000000G | EF | | 01 | FB | 00065 | CALLS | #1,INSERT_IN_ORDER | |
| 00000000G | EF | | 00 | FB | 0006C | CALLS | #0,MAKE_SCRATCH | : 1836 |
| | 50 | 00000000G | EF | DO | 00073 | MOVL | DEF_SCRATCH,R0 | : 1838 |
| 19 | AO | | 08 | 90 | 0007A | MOVB | #8,25(R0) | : 1845 |
| 1E | AO | 49 | 8F | 90 | 0007E | MOVB | #73,30(R0) | : 1846 |
| | | 00000000 | 8F | DF | 00083 | PUSHAL | #0 | : 1848 |
| 00000000G | EF | | 01 | FB | 00089 | CALLS | #1,INSERT_IN_ORDER | |
| 00000000G | EF | | 00 | FB | 00090 | CALLS | #0,MAKE_SCRATCH | : 1852 |
| | 50 | 00000000G | EF | DO | 00097 | MOVL | DEF_SCRATCH,R0 | : 1854 |
| 19 | AO | | 08 | 90 | 0009E | MOVB | #8,25(R0) | : 1861 |
| 1E | AO | 54 | 8F | 90 | 000A2 | MOVB | #84,30(R0) | : 1862 |
| 27 | AO | | 0A | C7 | 000A7 | DIVL3 | #10,ALLOC,39(R0) | : 1863 |
| | 5C | | 8F | DF | 000AC | PUSHAL | #0 | : 1865 |
| | | 00000000 | 01 | FB | 000B2 | CALLS | #1,INSERT_IN_ORDER | |
| 00000000G | EF | | 04 | 000B9 | RET | | | : 1869 |

; Routine Size: 186 bytes, Routine Base: \$CODE + 01023

| | | | | | | | | |
|--------------|----|-----------|------|-------|----------|--------|---------------------------|--------|
| | | | 001C | 00000 | REL_DEF: | WORD | *M<R2,R3,R4> | : 1916 |
| | | 5E | 08 | C2 | 00002 | SUBL2 | #8,SP | |
| | | 000000020 | 8F | DF | 00005 | PUSHAL | #32 | : 1931 |
| | | | 01 | FB | 0000B | CALLS | #1,QUERY | |
| 00000000G | EF | | 01 | C1 | 00012 | ADDL3 | #1,IData+228,RECORD_TOT | : 1937 |
| 5C 000000E4G | EF | | 00 | E1 | 0001A | BBC | #0,VARIABLE_RECORDS,3\$ | : 1939 |
| 00V00000000G | EF | | 02 | C0 | 00022 | ADDL2 | #2,RECORD_TOT | : 1941 |
| 50 | 5C | | 10 | C5 | 00025 | MULL3 | #16,RECORD_TOT,BUCKET_TOT | : 1943 |
| 52 | 50 | 00000200 | 8F | C7 | 00029 | DIVL3 | #512,BUCKET_TOT,BUCKET | : 1945 |
| | 01 | | 52 | D1 | 00031 | CMPL | BUCKET,#1 | : 1947 |
| | | | 00V | 18 | 00034 | BGEQ | 5\$ | |
| | 52 | | 01 | DO | 00036 | MOVL | #1,BUCKET | : 1949 |
| 50 | 00 | | 00 | 7A | 00039 | EMUL | #0,#0,BUCKET_TOT,R0 | : 1951 |
| 50 | 50 | 00000200 | 8F | 7B | 0003E | EDIV | #512,R0,R0,R0 | |
| | | | 50 | D5 | 00047 | TSTL | R0 | |
| | | | 00V | 18 | 00049 | BGEQ | 6\$ | |
| | 50 | 00000200 | 8F | C0 | 0004B | ADDL2 | #512,R0 | |
| | | | 50 | D5 | 00052 | TSTL | R0 | |
| | | | 00V | 13 | 00054 | BEQL | 8\$ | |
| | | 0000003F | 52 | D6 | 00056 | INCL | BUCKET | : 1953 |
| | FC | AD | 8F | DF | 00058 | PUSHAL | #63 | : 1955 |
| | | FC | 52 | DO | 0005E | MOVL | BUCKET,-4(FP) | |
| | | 00000080G | AD | 9F | 00062 | PUSHAB | -4(FP) | |
| 00000000G | EF | | 03 | FB | 00065 | PUSHAB | IData+128 | |
| | 52 | | 50 | DO | 00072 | CALLS | #3,MAX_FACTOR | |
| 50 | 52 | 00000200 | 8F | C5 | 00075 | MOVL | R0,BUCKET | : 1958 |
| | 50 | | 5C | C6 | 0007D | MULL3 | #512,BUCKET,R0 | |
| | 01 | | 50 | D1 | 00080 | DIVL2 | RECORD_TOT,R0 | : 1960 |
| | | | | | | CMPL | RECS_PER_BUCKET,#1 | |

| | | | | | | | | | |
|----|-----------|-----------|-----|-------|-------|--------|---|---|------|
| | | | 00V | 18 | 00083 | BGEQ | 10\$ | | |
| | | | 01 | D0 | 00085 | MOVL | #1, RECS PER BUCKET | : | 1962 |
| | 5C | 000000C0G | EF | C7 | 00088 | DIVL3 | RECS_PER_BUCKET, IDATA+192, NUM_BUCKETS | : | 1964 |
| | | | 01 | D1 | 00090 | CMPL | NUM_BUCKETS, #1 | : | 1966 |
| | | | 00V | 18 | 00093 | BGEQ | 12\$ | | |
| | | | 01 | D0 | 00095 | MOVL | #1, NUM_BUCKETS | : | 1968 |
| 53 | 000000C0G | EF | 00 | 7A | 00098 | EMUL | #0, #0, IDATA+192, R3 | : | 1970 |
| 53 | | 53 | 50 | 7B | 000A1 | EDIV | RECS_PER_BUCKET, R3, R3, R3 | | |
| | | | 53 | D5 | 000A6 | TSTL | R3 | | |
| | | | 00V | 18 | 000A8 | BGEQ | 13\$ | | |
| | | | 50 | C0 | 000AA | ADDL2 | RECS_PER_BUCKET, R3 | | |
| | | | 53 | D5 | 000AD | TSTL | R3 | | |
| | | | 00V | 13 | 000AF | BEQL | 15\$ | | |
| | | | 5C | D6 | 000B1 | INCL | NUM_BUCKETS | : | 1972 |
| | | | 52 | C4 | 000B3 | MULL2 | BUCKET, NUM_BUCKETS | : | 1977 |
| | | | EF | C0 | 000B6 | ADDL2 | IDATA+128, NUM_BUCKETS | | |
| | 00000000G | EF | 00 | FB | 000BD | CALLS | #0, MAKE SCRATCH | : | 1982 |
| | | | 50 | D0 | 000C4 | MOVL | DEF_SCRATCH, R0 | : | 1984 |
| | 19 | A0 | 08 | 90 | 000CB | MOVB | #8, 25(R0) | : | 1991 |
| | 1E | A0 | 8F | 90 | 000CF | MOVB | #72, 30(R0) | : | 1992 |
| | 27 | A0 | 5C | D0 | 000D4 | MOVL | ALLOC, 39(R0) | : | 1993 |
| | | | 8F | DF | 000D8 | PUSHAL | #0 | : | 1995 |
| | 00000000G | EF | 01 | FB | 000DE | CALLS | #1, INSERT IN ORDER | | |
| | 00000000G | EF | 00 | FB | 000E5 | CALLS | #0, MAKE SCRATCH | : | 1999 |
| | | | 50 | D0 | 000EC | MOVL | DEF_SCRATCH, R0 | : | 2001 |
| | 19 | A0 | 08 | 90 | 000F3 | MOVB | #8, 25(R0) | : | 2008 |
| | 1E | A0 | 8F | 90 | 000F7 | MOVB | #73, 30(R0) | : | 2009 |
| | | | 8F | DF | 000FC | PUSHAL | #0 | : | 2011 |
| | 00000000G | EF | 01 | FB | 00102 | CALLS | #1, INSERT IN ORDER | | |
| | 00000000G | EF | 00 | FB | 00109 | CALLS | #0, MAKE SCRATCH | : | 2015 |
| | | | 50 | D0 | 00110 | MOVL | DEF_SCRATCH, R0 | : | 2017 |
| | 19 | A0 | 08 | 90 | 00117 | MOVB | #8, 25(R0) | : | 2024 |
| | 1E | A0 | 8F | 90 | 0011B | MOVB | #74, 30(R0) | : | 2025 |
| | 27 | A0 | 52 | D0 | 00120 | MOVL | BUCKET, 39(R0) | : | 2026 |
| | | | 8F | DF | 00124 | PUSHAL | #0 | : | 2028 |
| | 00000000G | EF | 01 | FB | 0012A | CALLS | #1, INSERT IN ORDER | | |
| | 00000000G | EF | 00 | FB | 00131 | CALLS | #0, MAKE SCRATCH | : | 2032 |
| | | | 53 | D0 | 00138 | MOVL | DEF_SCRATCH, R3 | : | 2034 |
| | 19 | A3 | 08 | 90 | 0013F | MOVB | #8, 25(R3) | : | 2041 |
| | 1E | A3 | 8F | 90 | 00143 | MOVB | #84, 30(R3) | : | 2042 |
| | | | 8F | DF | 00148 | PUSHAL | #999999999 | : | 2043 |
| FC | AD | | 04 | C7 | 0014E | DIVL3 | #4, ALLOC, -4(FP) | | |
| | | | AD | 9F | 00153 | PUSHAB | -4(FP) | | |
| | F8 | AD | 52 | D0 | 00156 | MOVL | BUCKET, -8(FP) | | |
| | | | AD | 9F | 0015A | PUSHAB | -8(FP) | | |
| | 00000000G | EF | 03 | FB | 0015D | CALLS | #3, MAX_FACTOR | | |
| | 27 | A3 | 50 | D0 | 00164 | MOVL | R0, 39(R3) | | |
| | | | 8F | DF | 00168 | PUSHAL | #0 | : | 2048 |
| | 00000000G | EF | 01 | FB | 0016E | CALLS | #1, INSERT IN ORDER | | |
| | 00000000G | EF | 00 | FB | 00175 | CALLS | #0, MAKE SCRATCH | : | 2052 |
| | | | 50 | D0 | 0017C | MOVL | DEF_SCRATCH, R0 | : | 2054 |
| | 19 | A0 | 08 | 90 | 00183 | MOVB | #8, 25(R0) | : | 2061 |
| | 1E | A0 | 8F | 90 | 00187 | MOVB | #92, 30(R0) | : | 2062 |
| | 27 | A0 | EF | D0 | 0018C | MOVL | IDATA+192, 39(R0) | : | 2063 |
| | | | 8F | DF | 00194 | PUSHAL | #0 | : | 2065 |
| | 00000000G | EF | 01 | FB | 0019A | CALLS | #1, INSERT_IN_ORDER | | |
| | | | 04 | 001A1 | RET | | | : | 2069 |

: Routine Size: 418 bytes, Routine Base: \$CODE + 010DD

| | | | | | | |
|-----------|------|-------------|--------------|-------------|---------------------------------------|--------|
| | | | 00000 | APPEND_DEF: | | : 2117 |
| | | | 00FC 00000 | .WORD | ^M<R2,R3,R4,R5,R6,R7> | |
| | SE | | 04 C2 00002 | SUBL2 | #4,SP | |
| | 05 | 00000118G | EF D1 00005 | CMPL | IDATA+280,#5 | : 2144 |
| | | | 00V 13 0000C | BEQL | 13\$ | |
| 04 | 00 | 00000118G | EF CF 0000E | CASEL | IDATA+280,#0,#4 | : 2146 |
| | | | 0000V 00016 | .DISPL | 2\$ | |
| | | | 0000V 00018 | .DISPL | 9\$ | |
| | | | 0000V 0001A | .DISPL | 4\$ | |
| | | | 0000V 0001C | .DISPL | 6\$ | |
| | | | 0000V 0001E | .DISPL | 8\$ | |
| | | | 00V 11 00020 | BRB | 11\$ | |
| | | 0000002B | 8F DF 00022 | PUSHAL | #43 | : 2148 |
| 00000000G | EF | | 01 FB 00028 | CALLS | #1,QUERY | |
| | | | 00V 11 0002F | BRB | 13\$ | |
| | | 00000030 | 8F DF 00031 | PUSHAL | #48 | : 2150 |
| 00000000G | EF | | 01 FB 00037 | CALLS | #1,QUERY | |
| | | | 00V 11 0003E | BRB | 13\$ | |
| | | 00000022 | 8F DF 00040 | PUSHAL | #34 | : 2152 |
| 00000000G | EF | | 01 FB 00046 | CALLS | #1,QUERY | |
| | | | 00V 11 0004D | BRB | 13\$ | |
| | | 00000000G | 00 FB 0004F | CALLS | #0,ASK_KEY_SIZE | : 2154 |
| | | | 00V 11 00056 | BRB | 13\$ | |
| 00000000G | EF | | 00 FB 00058 | CALLS | #0,ASK_MEAN_RECORD_SIZE | : 2160 |
| 00000000G | EF | | 00 FB 0005F | CALLS | #0,MAKE_SCRATCH | : 2165 |
| | 50 | 00000000G | EF D0 00066 | MOVL | DEF_SCRATCH,R0 | : 2167 |
| | 19 | | 0C 90 0006D | MOVB | #12,25(R0) | : 2171 |
| | 1E | | 8F 90 00071 | MOVB | #-116,30(R0) | : 2172 |
| | 27 | | EF D0 00076 | MOVL | IDATA+228,39(R0) | : 2173 |
| | | 000000000 | 8F DF 0007E | PUSHAL | #0 | : 2175 |
| 00000000G | EF | | 01 FB 00084 | CALLS | #1,INSERT_IN_ORDER | |
| | | | 00V 11 0008B | BRB | 13\$ | |
| | | | 0008D | | | |
| | | 0658 | 00 FB 0008D | CALLS | #0,NATURAL_DEPTH | : 2193 |
| 00000000G | EF | | 50 D0 00092 | MOVL | R0,BUCKET_DEFAULT | |
| | | 00000025 | 8F DF 00099 | PUSHAL | #37 | : 2195 |
| 00000000G | EF | | 01 FB 0009F | CALLS | #1,QUERY | |
| | | | 50 D4 000A6 | CLRL | R0 | : 2197 |
| | 5C | | 50 D0 000A8 | MOVL | R0,1 | |
| | | 00000000GEF | 4C D4 000AB | CLRL | INIT_PRIMARY_BUCKETS[1] | : 2201 |
| | | 00000000GEF | 4C D4 000B2 | CLRL | ADDED_PRIMARY_BUCKETS[1] | : 2202 |
| EB | | | 1F F3 000B9 | AOBLEQ | #31,R0,15\$ | |
| | 04ED | | 00 FB 000BD | CALLS | #0,PROLOGUE3_DEPTH | : 2206 |
| | | | 50 D0 000C2 | MOVL | R0,CHOSEN_DEPTH | |
| | | 0000001C | 8F DF 000C5 | PUSHAL | #28 | : 2211 |
| 00000000G | EF | | 01 FB 000CB | CALLS | #1,QUERY | |
| | | 0000000D | 8F DF 000D2 | PUSHAL | #13 | : 2212 |
| 00000000G | EF | | 01 FB 000D8 | CALLS | #1,QUERY | |
| | | 00000000G | EF D0 000DF | MOVL | INIT_NUMBER_BUCKETS,INIT_DATA_ALLOC | : 2217 |
| | 51 | 00000000G | EF D0 000E6 | MOVL | ADDED_NUMBER_BUCKETS,ADDED_DATA_ALLOC | : 2218 |
| | | | 53 D4 000ED | CLRL | INIT_INDEX_ALLOC | : 2223 |
| | | | 54 D4 000EF | CLRL | ADDED_INDEX_ALLOC | : 2224 |
| | | | 01 D0 000F1 | MOVL | #1,R5 | : 2226 |
| | 56 | | 52 D0 000F4 | MOVL | CHOSEN_DEPTH,R6 | |

| | | | | | | |
|----------------|----------|-----------|-------|-------|---|--|
| 56 | 55 | D1 | 000F7 | CMPL | R5,R6 | |
| | 00V | 15 | 000FA | BLEQ | 19\$ | |
| | 00V | 11 | 000FC | BRB | 20\$ | |
| | 55 | D6 | 000FE | 18\$: | INCL | R5 |
| 5C | 55 | D0 | 00100 | 19\$: | MOVL | R5,I |
| 53 00000000GEF | 4C | C0 | 00103 | ADDL2 | INIT_NUMBER_BUCKETS[I],INIT_INDEX_ALLOC | : 2230 |
| 54 00000000GEF | 4C | C0 | 0010B | ADDL2 | ADDED_NUMBER_BUCKETS[I],ADDED_INDEX_ALLOC | : 2231 |
| 56 | 55 | D1 | 00113 | CMPL | R5,R6 | |
| | E6 | 19 | 00116 | BLSS | 18\$ | |
| 00000088G | EF | D5 | 00118 | 20\$: | TSTL | IDATA+136 |
| | 00V | 13 | 0011E | BEQL | 32\$ | : 2238 |
| 55 | 50 | 4E | 00120 | CVTLF | INIT_DATA_ALLOC,R5 | : 2242 |
| 55 00000004G | EF | 44 | 00123 | MULF2 | RDATA+4,R5 | |
| 55 | 55 | 4A | 0012A | CVTFL | R5,R5 | |
| | 55 | D6 | 0012D | INCL | R5 | |
| 56 | 53 | 4E | 0012F | CVTLF | INIT_INDEX_ALLOC,R6 | : 2244 |
| 56 00000004G | EF | 44 | 00132 | MULF2 | RDATA+4,R6 | |
| 56 | 56 | 4A | 00139 | CVTFL | R6,R6 | |
| | 56 | D6 | 0013C | INCL | R6 | |
| 55 | 50 | 55 | C3 | 0013E | SUBL3 | USED_DATA_BUCKETS,INIT_DATA_ALLOC,- |
| | | | | 00142 | | UNUSED_DATA_BUCKETS |
| 56 | 53 | 56 | C3 | 00142 | SUBL3 | USED_INDEX_BUCKETS,INIT_INDEX_ALLOC,- |
| | | | | 00146 | | UNUSED_INDEX_BUCKETS |
| 55 | 51 | D1 | 00146 | CMPL | ADDED_DATA_ALLOC,UNUSED_DATA_BUCKETS | : 2249 |
| | 00V | 15 | 00149 | BLEQ | 23\$ | |
| 51 | 55 | C2 | 0014B | SUBL2 | UNUSED_DATA_BUCKETS,ADDED_DATA_ALLOC | : 2251 |
| | 00V | 11 | 0014E | BRB | 24\$ | |
| | 51 | D4 | 00150 | 23\$: | CLRL | ADDED_DATA_ALLOC |
| 56 | 54 | D1 | 00152 | 24\$: | CMPL | ADDED_INDEX_ALLOC,UNUSED_INDEX_BUCKETS |
| | 00V | 15 | 00155 | BLEQ | 26\$ | |
| 54 | 56 | C2 | 00157 | SUBL2 | UNUSED_INDEX_BUCKETS,ADDED_INDEX_ALLOC | : 2259 |
| | 00V | 11 | 0015A | BRB | 27\$ | |
| | 54 | D4 | 0015C | 26\$: | CLRL | ADDED_INDEX_ALLOC |
| | 51 | D5 | 0015E | 27\$: | TSTL | ADDED_DATA_ALLOC |
| | 00V | 15 | 00160 | BLEQ | 29\$ | |
| 50 | 51 | C0 | 00162 | ADDL2 | ADDED_DATA_ALLOC,INIT_DATA_ALLOC | : 2267 |
| | 54 | D5 | 00165 | 29\$: | TSTL | ADDED_INDEX_ALLOC |
| | 00V | 15 | 00167 | BLEQ | 32\$ | |
| 54 3B9AC9FF | 53 | 54 | C0 | 00169 | ADDL2 | ADDED_INDEX_ALLOC,INIT_INDEX_ALLOC |
| 8F 00000094G | EF | C7 | 0016C | 32\$: | DIVL3 | IDATA+148,#999999999,R5 |
| 54 | 50 | D1 | 00178 | CMPL | INIT_DATA_ALLOC,R4 | : 2280 |
| | 00V | 15 | 0017B | BLEQ | 34\$ | |
| 56 3B9AC9FF | 8F | D0 | 0017D | MOVL | #999999999,DATA_ALLOC | : 2282 |
| | 00V | 11 | 00184 | BRB | 35\$ | |
| 56 | 50 | 00000094G | EF | C5 | 00186 | 34\$: |
| | 54 | 53 | D1 | 0018E | 35\$: | MULL3 |
| | | 00V | 15 | 00191 | CMPL | IDATA+148,INIT_DATA_ALLOC,DATA_ALLOC |
| | 54 | 3B9AC9FF | 8F | D0 | 00193 | INIT_INDEX_ALLOC,R4 |
| | | 00V | 11 | 0019A | BLEQ | 37\$ |
| 54 00000000G | 53 | 00000094G | EF | C5 | 0019C | 37\$: |
| | EF | 00 | FB | 001A4 | 38\$: | MULL3 |
| | 00000000 | 8F | DF | 001AB | CALLS | IDATA+148,INIT_INDEX_ALLOC,INDEX_ALLOC |
| | 00 | 8F | 9F | 001B1 | PUSHAL | #0,POINT_AT_DEFINITION |
| | 00000000 | 8F | DF | 001B4 | PUSHAB | #0 |
| | 08 | 8F | 9F | 001BA | PUSHAB | #0 |
| | 00 | 8F | 9F | 001BD | PUSHAB | #8 |
| 00000000G | EF | 05 | FB | 001C0 | CALLS | #5,FIND_OBJECT |

| | | | | | | | | | |
|-----------|-------------|-----------|-----|-----|-------|--------|---------------------------------------|-------------------------------|--------|
| 53 | 00000000G | 00V | 50 | E9 | 001C7 | BLBC | R0,50\$ | | |
| | | EF | 19 | C1 | 001CA | ADDL3 | #25,DEF_CURRENT,R3 | : | 2307 |
| | | 08 | 63 | 91 | 001D2 | CMPB | (R3),#8 | : | 2309 |
| | | | 00V | 12 | 001D5 | BNEQ | 43\$ | | |
| | | 50 | EF | D0 | 001D7 | MOVL | DEF_CURRENT,R0 | | |
| | | 50 | A0 | 9A | 001DE | MOVZBL | 30(R0),R0 | | |
| | 00000098 | 8F | 50 | D1 | 001E2 | CMPL | R0,#152 | | |
| | | | 00V | 1E | 001E9 | BGEQU | 43\$ | | |
| | 00VFFFEC A2 | EF | 50 | E1 | 001EB | BBC | R0,C.AAI,43\$ | | |
| | 00000000G | EF | 00 | FB | 001F3 | CALLS | #0,DELETE_CURRENT | : | 2316 |
| | | | 00V | 11 | 001FA | BRB | 44\$ | | |
| | 00000000G | EF | 00 | FB | 001FC | CALLS | #0,INCR_CURRENT | : | 2320 |
| | | | 50 | 94 | 00203 | CLRB | R0 | | |
| | | 00000000G | EF | D5 | 00205 | TSTL | DEF_CURRENT | | |
| | | | 00V | 12 | 0020B | BNEQ | 46\$ | | |
| | | | 50 | 96 | 0020D | INCB | R0 | | |
| | | | 51 | 94 | 0020F | CLRB | R1 | | |
| | 53 | 000000G3G | EF | 19 | C1 | 00211 | ADDL3 | #25,DEF_CURRENT,R3 | |
| | | | 08 | 63 | 91 | 00219 | CMPB | (R3),#8 | |
| | | | | 00V | 13 | 0021C | BEQL | 48\$ | |
| | | | 51 | 96 | 0021E | INCB | R1 | | |
| | | | 50 | 88 | 00220 | BISB2 | R0,R1 | | |
| | | | 51 | E9 | 00223 | BLBC | R1,40\$ | | |
| 00000084G | EF | 7F | 8F | 00 | ED | 00226 | CMPZV | #0,#7,#^X7F,IData+132 | : 2329 |
| | | | | 00V | 15 | 00230 | BLEQ | 52\$ | |
| | 53 | 00000084G | EF | 02 | C5 | 00232 | MULL3 | #2,IData+132,DATA_AREA_NUMBER | : 2331 |
| | | | | 00V | 11 | 0023A | BRB | 53\$ | |
| | | | 53 | 9A | 0023C | MOVZBL | #254,DATA_AREA_NUMBER | : 2335 | |
| | | | 53 | C1 | 00240 | ADDL3 | #1,DATA_AREA_NUMBER,INDEX_AREA_NUMBER | : 2337 | |
| | 55 | 00000000G | EF | 00 | FB | 00244 | CALLS | #0,MAKE_SCRATCH | : 2342 |
| | | | 50 | D0 | 0024B | MOVL | DEF_SCRATCH,R0 | : 2344 | |
| | | | | 60 | 94 | 00252 | CLRB | (R0) | : 2351 |
| | | | 19 | 05 | 90 | 00254 | MOVB | #5,25(R0) | : 2352 |
| | | | 1A | 53 | D0 | 00258 | MOVL | DATA_AREA_NUMBER,26(R0) | : 2353 |
| | | | | 8F | DF | 0025C | PUSHAL | #0 | : 2355 |
| | 00000000G | EF | 01 | FB | 00262 | CALLS | #1,INSERT_IN_ORDER | | |
| | | | 00 | ED | 00269 | CMPZV | #0,#7,#^X7F,IData+132 | : 2362 | |
| | | | 00V | 15 | 00273 | BLEQ | 56\$ | | |
| | | | 5C | D4 | 00275 | CLRL | TEMP_ALLOC | : 2364 | |
| | | | 00V | 11 | 00277 | BRB | 60\$ | | |
| | | 00000000 | 8F | DF | 00279 | PUSHAL | #0 | : 2366 | |
| | | 1B | 8F | 9F | 0027F | PUSHAB | #27 | | |
| | | 000000FE | 8F | DF | 00282 | PUSHAL | #254 | | |
| | | 05 | 8F | 9F | 00288 | PUSHAB | #5 | | |
| | | 01 | 8F | 9F | 0028B | PUSHAB | #1 | | |
| | 00000000G | EF | 05 | FB | 0028E | CALLS | #5,FIND_OBJECT | | |
| | | 00V | 50 | E9 | 00295 | BLBC | R0,58\$ | | |
| | | 50 | EF | D0 | 00298 | MOVL | DEF_CURRENT,R0 | : 2368 | |
| | | 5C | A0 | D0 | 0029F | MOVL | 39(R0),TEMP_ALLOC | | |
| | | | 00V | 11 | 002A3 | BRB | 60\$ | | |
| | | | 5C | D4 | 002A5 | CLRL | TEMP_ALLOC | : 2372 | |
| | 00000000G | EF | 00 | FB | 002A7 | CALLS | #0,MAKE_SCRATCH | : 2374 | |
| | | | 50 | D0 | 002AE | MOVL | DEF_SCRATCH,R0 | : 2376 | |
| | | | 19 | 05 | 90 | MOVB | #5,25(R0) | : 2383 | |
| | | | 1A | 53 | D0 | MOVL | DATA_AREA_NUMBER,26(R0) | : 2384 | |
| | | | 1E | 1B | 90 | MOVB | #27,30(R0) | : 2385 | |
| | | | 56 | 5C | C0 | ADDL2 | TEMP_ALLOC,DATA_ALLOC | : 2387 | |

Generated Code

C 12
16-Sep-1984 01:10:30
5-Sep-1984 13:36:36VAX-11 Pascal V2.4-277
DISK\$VMSMASTER:[EDF.SRC]EDFDESIGN.PAS;1 (38)

Page 107

| | | | | | | | | | |
|-----------|-----|-----------|-----|-------|--------|--------|--------------------------|---|------|
| 27 | A0 | 00000000 | 56 | D0 | 002C4 | MOVL | DATA_ALLOC,39(R0) | | |
| 00000000G | EF | | 8F | DF | 002C8 | PUSHAL | #0 | : | 2389 |
| 00000000G | EF | | 01 | FB | 002CE | CALLS | #1,INSERT IN ORDER | | |
| | 50 | 00000000G | 00 | FB | 002D5 | CALLS | #0,MAKE SCRATCH | : | 2393 |
| 19 | A0 | | EF | D0 | 002DC | MOVL | DEF_SCRATCH,R0 | : | 2395 |
| 1A | A0 | | 05 | 90 | 002E3 | MOVB | #5,25(R0) | : | 2402 |
| 1E | A0 | | 53 | D0 | 002E7 | MOVL | DATA_AREA_NUMBER,26(R0) | : | 2403 |
| | | 00000000 | 1C | 90 | 002EB | MOVB | #28,30(R0) | : | 2404 |
| 00000000G | EF | | 8F | DF | 002EF | PUSHAL | #0 | : | 2406 |
| 00000000G | EF | | 01 | FB | 002F5 | CALLS | #1,INSERT IN ORDER | | |
| | 50 | 00000000G | 00 | FB | 002FC | CALLS | #0,MAKE SCRATCH | : | 2410 |
| 19 | A0 | | EF | D0 | 00303 | MOVL | DEF_SCRATCH,R0 | : | 2412 |
| 1A | A0 | | 05 | 90 | 0030A | MOVB | #5,25(R0) | : | 2419 |
| 1E | A0 | | 53 | D0 | 0030E | MOVL | DATA_AREA_NUMBER,26(R0) | : | 2420 |
| 27 | A0 | 00000094G | 1D | 90 | 00312 | MOVB | #29,30(R0) | : | 2421 |
| | | 00000000 | EF | D0 | 00316 | MOVL | IDATA+148,39(R0) | : | 2422 |
| 00000000G | EF | | 8F | DF | 0031E | PUSHAL | #0 | : | 2424 |
| 00000000G | EF | | 01 | FB | 00324 | CALLS | #1,INSERT IN ORDER | | |
| | 57 | 00000000G | 00 | FB | 0032B | CALLS | #0,MAKE SCRATCH | : | 2428 |
| 19 | A7 | | EF | D0 | 00332 | MOVL | DEF_SCRATCH,R7 | : | 2430 |
| 1A | A7 | | 05 | 90 | 00339 | MOVB | #5,25(R7) | : | 2437 |
| 1E | A7 | | 53 | D0 | 0033D | MOVL | DATA_AREA_NUMBER,26(R7) | : | 2438 |
| | | 3B9AC9FF | 20 | 90 | 00341 | MOVB | #32,30(R7) | : | 2439 |
| FC | AD | | 8F | DF | 00345 | PUSHAL | #999999999 | : | 2440 |
| | | FC | 04 | C7 | 0034B | DIVL3 | #4,R6,-4(FP) | | |
| | | 00000094G | 9F | 00350 | PUSHAB | -4(FP) | | | |
| 00000000G | EF | | EF | 9F | 00353 | PUSHAB | IDATA+148 | | |
| 27 | A7 | | 03 | FB | 00359 | CALLS | #3,MAX FACTOR | | |
| | | 00000000 | 50 | D0 | 00360 | MOVL | R0,39(R7) | | |
| 00000000G | EF | | 8F | DF | 00364 | PUSHAL | #0 | : | 2445 |
| 00000000G | EF | | 01 | FB | 0036A | CALLS | #1,INSERT IN ORDER | | |
| | 50 | 00000000G | 00 | FB | 00371 | CALLS | #0,MAKE SCRATCH | : | 2449 |
| | | | EF | D0 | 00378 | MOVL | DEF_SCRATCH,R0 | : | 2451 |
| 19 | A0 | | 60 | 94 | 0037F | CLRB | (R0) | : | 2458 |
| 1A | A0 | | 05 | 90 | 00381 | MOVB | #5,25(R0) | : | 2459 |
| | | 00000000 | 55 | D0 | 00385 | MOVL | INDEX_AREA_NUMBER,26(R0) | : | 2460 |
| 00000000G | EF | | 8F | DF | 00389 | PUSHAL | #0 | : | 2462 |
| 00000000G | EF | | 01 | FB | 0038F | CALLS | #1,INSERT IN ORDER | | |
| | 07 | | 00 | FB | 00396 | CALLS | #0,MAKE SCRATCH | : | 2466 |
| 00000084G | EF | | 00 | ED | 0039D | CMPZV | #0,#7,#*X7F,IDATA+132 | : | 2468 |
| | | | 00V | 15 | 003A7 | BLEQ | 67\$ | | |
| | | | 5C | D4 | 003A9 | CLRL | TEMP_ALLOC | : | 2470 |
| | | | 00V | 11 | 003AB | BRB | 71\$ | | |
| | | 00000000 | 8F | DF | 003AD | PUSHAL | #0 | : | 2472 |
| | | 18 | 8F | 9F | 003B3 | PUSHAB | #27 | | |
| | | 000000FF | 8F | DF | 003B6 | PUSHAL | #255 | | |
| | | 05 | 8F | 9F | 003BC | PUSHAB | #5 | | |
| | | 01 | 8F | 9F | 003BF | PUSHAB | #1 | | |
| 00000000G | EF | | 05 | FB | 003C2 | CALLS | #5,FIND_OBJECT | | |
| | 00V | | 50 | E9 | 003C9 | BLBC | R0,69\$ | | |
| | 50 | 00000000G | EF | D0 | 003CC | MOVL | DEF_CURRENT,R0 | : | 2474 |
| | 5C | 27 | A0 | D0 | 003D3 | MOVL | 39(R0),TEMP_ALLOC | | |
| | | | 00V | 11 | 003D7 | BRB | 71\$ | | |
| | | | 5C | D4 | 003D9 | CLRL | TEMP_ALLOC | : | 2478 |
| | | 00000000G | EF | D0 | 003DB | MOVL | DEF_SCRATCH,R0 | : | 2480 |
| 19 | A0 | | 05 | 90 | 003E2 | MOVB | #5,25(R0) | : | 2487 |
| 1A | A0 | | 55 | D0 | 003E6 | MOVL | INDEX_AREA_NUMBER,26(R0) | : | 2488 |

| | | | | | | | |
|-----------|----------|----|----|-------|--------|--------------------------|--------|
| 1E | A0 | 1B | 90 | 003EA | MOVB | #27,30(R0) | : 2489 |
| | SC | 54 | CO | 003EE | ADDL2 | INDEX_ALLOC,TEMP_ALLOC | : 2490 |
| 27 | A0 | 5C | DO | 003F1 | MOVL | TEMP_ALLOC,39(R0) | |
| | | 8F | DF | 003F5 | PUSHAL | #0 | : 2492 |
| 00000000G | EF | 01 | FB | 003FB | CALLS | #1,INSERT IN ORDER | |
| 00000000G | EF | 00 | FB | 00402 | CALLS | #0,MAKE SCRATCH | : 2496 |
| | 50 | EF | DO | 00409 | MOVL | DEF_SCRATCH,R0 | : 2498 |
| 19 | A0 | 05 | 90 | 00410 | MOVB | #5,25(R0) | : 2505 |
| 1A | A0 | 55 | DO | 00414 | MOVL | INDEX_AREA_NUMBER,26(R0) | : 2506 |
| 1E | A0 | 1C | 90 | 00418 | MOVB | #28,30(R0) | : 2507 |
| | | 8F | DF | 0041C | PUSHAL | #0 | : 2509 |
| 00000000G | EF | 01 | FB | 00422 | CALLS | #1,INSERT IN ORDER | |
| 00000000G | EF | 00 | FB | 00429 | CALLS | #0,MAKE SCRATCH | : 2513 |
| | 50 | EF | DO | 00430 | MOVL | DEF_SCRATCH,R0 | : 2515 |
| 19 | A0 | 05 | 90 | 00437 | MOVB | #5,25(R0) | : 2522 |
| 1A | A0 | 55 | DO | 0043B | MOVL | INDEX_AREA_NUMBER,26(R0) | : 2523 |
| 1E | A0 | 1D | 90 | 0043F | MOVB | #29,30(R0) | : 2524 |
| 27 | A0 | EF | DO | 00443 | MOVL | IDATA+148,39(R0) | : 2525 |
| | | 8F | DF | 0044B | PUSHAL | #0 | : 2527 |
| 00000000G | EF | 01 | FB | 00451 | CALLS | #1,INSERT IN ORDER | |
| 00000000G | EF | 00 | FB | 00458 | CALLS | #0,MAKE SCRATCH | : 2531 |
| | 54 | EF | DO | 0045F | MOVL | DEF_SCRATCH,R4 | : 2533 |
| 19 | A4 | 05 | 90 | 00466 | MOVB | #5,25(R4) | : 2540 |
| 1A | A4 | 55 | DO | 0046A | MOVL | INDEX_AREA_NUMBER,26(R4) | : 2541 |
| 1E | A4 | 20 | 90 | 0046E | MOVB | #32,30(R4) | : 2542 |
| | 3B9AC9FF | 8F | DF | 00472 | PUSHAL | #999999999 | : 2543 |
| FC | AD | 04 | C7 | 00478 | DIVL3 | #4,R12,-4(FP) | |
| | | 9F | 9F | 0047D | PUSHAB | -4(FP) | |
| | | EF | 9F | 00480 | PUSHAB | IDATA+148 | |
| 00000000G | EF | 03 | FB | 00486 | CALLS | #3,MAX FACTOR | |
| 27 | A4 | 50 | DO | 0048D | MOVL | R0,39(R4) | |
| | | 8F | DF | 00491 | PUSHAL | #0 | : 2548 |
| 00000000G | EF | 01 | FB | 00497 | CALLS | #1,INSERT IN ORDER | |
| 00000000G | EF | 00 | FB | 0049E | CALLS | #0,MAKE SCRATCH | : 2552 |
| | 50 | EF | DO | 004A5 | MOVL | DEF_SCRATCH,R0 | : 2554 |
| 1A | A0 | 60 | 94 | 004AC | CLRB | (R0) | : 2561 |
| | | EF | DO | 004AE | MOVL | IDATA+132,26(R0) | : 2562 |
| 00000000G | EF | 8F | DF | 004B6 | PUSHAL | #0 | : 2564 |
| 00000000G | EF | 01 | FB | 004BC | CALLS | #1,INSERT IN ORDER | |
| | 50 | 00 | FB | 004C3 | CALLS | #0,MAKE SCRATCH | : 2568 |
| 1A | A0 | EF | DO | 004CA | MOVL | DEF_SCRATCH,R0 | : 2570 |
| 1E | A0 | EF | DO | 004D1 | MOVL | IDATA+132,26(R0) | : 2577 |
| 2B | A0 | 8F | 90 | 004D9 | MOVB | #119,30(R0) | : 2578 |
| | | EF | 90 | 004DE | MOVB | BDATA+21,43(R0) | : 2579 |
| 00000000G | EF | 8F | DF | 004E6 | PUSHAL | #0 | : 2581 |
| 00000000G | EF | 01 | FB | 004EC | CALLS | #1,INSERT IN ORDER | |
| | 50 | 00 | FB | 004F3 | CALLS | #0,MAKE SCRATCH | : 2585 |
| 1A | A0 | EF | DO | 004FA | MOVL | DEF_SCRATCH,R0 | : 2587 |
| 1E | A0 | EF | DO | 00501 | MOVL | IDATA+132,26(R0) | : 2594 |
| 27 | A0 | 8F | 90 | 00509 | MOVB | #120,30(R0) | : 2595 |
| | | 53 | DO | 0050E | MOVL | DATA_AREA_NUMBER,39(R0) | : 2596 |
| 00000000G | EF | 8F | DF | 00512 | PUSHAL | #0 | : 2598 |
| 00000000G | EF | 01 | FB | 00518 | CALLS | #1,INSERT IN ORDER | |
| | 50 | 00 | FB | 0051F | CALLS | #0,MAKE SCRATCH | : 2602 |
| 1A | A0 | EF | DO | 00526 | MOVL | DEF_SCRATCH,R0 | : 2604 |
| 1E | A0 | EF | DO | 0052D | MOVL | IDATA+132,26(R0) | : 2611 |
| | | 8F | 90 | 00535 | MOVB | #121,30(R0) | : 2612 |

| Generated Code | | | |
|----------------|----|-----------|--------------------------------------|
| 27 | A0 | 00000000G | EF D0 0053A |
| | | 00000000 | 8F DF 00542 |
| 00000000G | EF | | 01 FB 00548 |
| 00000000G | EF | | 00 FB 0054F |
| | 50 | 00000000G | EF D0 00556 |
| 1A | A0 | 00000084G | EF D0 0055D |
| 1E | A0 | 7A | 8F 90 00565 |
| 2B | A0 | 0000000CG | EF 90 0056A |
| | | 00000000 | 8F DF 00572 |
| 00000000G | EF | | 01 FB 00578 |
| 00000000G | EF | | 00 FB 0057F |
| | | 00000084G | EF D5 00586 |
| | | | 00V 12 0058C |
| | 50 | 00000000G | EF D0 0058E |
| 1A | A0 | 00000084G | EF D0 00595 |
| 1E | A0 | 7B | 8F 90 0059D |
| 2B | A0 | 0000000DG | EF 90 005A2 |
| | | 00000000 | 8F DF 005AA |
| 00000000G | EF | | 01 FB 005B0 |
| 00000000G | EF | | 00 FB 005B7 |
| | 50 | 00000000G | EF D0 005BE 83\$: |
| 1A | A0 | 00000084G | EF D0 005C5 |
| 1E | A0 | 7C | 8F 90 005CD |
| 2B | A0 | 00000017G | EF 90 005D2 |
| | | 00000000 | 8F DF 005DA |
| 00000000G | EF | | 01 FB 005E0 |
| 00000000G | EF | | 00 FB 005E7 |
| | 50 | 00000000G | EF D0 005EE |
| 1A | A0 | 00000084G | EF D0 005F5 |
| 1E | A0 | 7D | 8F 90 005FD |
| 27 | A0 | | 55 D0 00602 |
| | | 00000000 | 8F DF 00606 |
| 00000000G | EF | | 01 FB 0060C |
| 00000000G | EF | | 00 FB 00613 |
| | 50 | 00000000G | EF D0 0061A |
| 1A | A0 | 00000084G | EF D0 00621 |
| 1E | A0 | 7F | 8F 90 00629 |
| 27 | A0 | 00000000G | EF D0 0062E |
| | | 00000000 | 8F DF 00636 |
| 00000000G | EF | | 01 FB 0063C |
| 00000000G | EF | | 00 FB 00643 |
| | 50 | 00000000G | EF D0 0064A |
| 1A | A0 | 00000084G | EF D0 00651 |
| 1E | A0 | 7E | 8F 90 00659 |
| 2B | A0 | 0000000EG | EF 90 0065E |
| | | 00000000 | 8F DF 00666 |
| 00000000G | EF | | 01 FB 0066C |
| 00V00000013G | EF | | 00 E0 00673 |
| 00000000G | EF | | 00 FB 0067B |
| | 50 | 00000000G | EF D0 00682 |
| 1A | A0 | 00000084G | EF D0 00689 |
| 1E | A0 | 85 | 8F 90 00691 |
| 27 | A0 | 000000D8G | EF D0 00696 |
| | | 00000000 | 8F DF 0069E |
| 00000000G | EF | | 01 FB 006A4 |
| | | | 00V 11 006AB |
| | | | 53 D4 006AD 90\$: |
| | | | MOVL IDATA,39(R0) : 2613 |
| | | | PUSHAL #0 : 2615 |
| | | | CALLS #1,INSERT_IN_ORDER : 2619 |
| | | | CALLS #0,MAKE_SCRATCH : 2621 |
| | | | MOVL DEF_SCRATCH,R0 : 2628 |
| | | | MOVL IDATA+132,26(R0) : 2629 |
| | | | MOVB #122,30(R0) : 2630 |
| | | | MOVB BDATA+12,43(R0) : 2632 |
| | | | PUSHAL #0 : 2636 |
| | | | CALLS #1,INSERT_IN_ORDER : 2638 |
| | | | CALLS #0,MAKE_SCRATCH : 2642 |
| | | | TSTL IDATA+132 : 2649 |
| | | | BNEQ 83\$: 2650 |
| | | | MOVL DEF_SCRATCH,R0 : 2651 |
| | | | MOVL IDATA+132,26(R0) : 2653 |
| | | | MOVB #123,30(R0) : 2657 |
| | | | MOVB BDATA+13,43(R0) : 2661 |
| | | | PUSHAL #0 : 2668 |
| | | | CALLS #1,INSERT_IN_ORDER : 2669 |
| | | | CALLS #0,MAKE_SCRATCH : 2670 |
| | | | MOVL DEF_SCRATCH,R0 : 2672 |
| | | | MOVL IDATA+132,26(R0) : 2676 |
| | | | MOVB #124,30(R0) : 2678 |
| | | | MOVB BDATA+23,43(R0) : 2685 |
| | | | PUSHAL #0 : 2686 |
| | | | CALLS #1,INSERT_IN_ORDER : 2687 |
| | | | CALLS #0,MAKE_SCRATCH : 2689 |
| | | | MOVL DEF_SCRATCH,R0 : 2693 |
| | | | MOVL IDATA+132,26(R0) : 2695 |
| | | | MOVB #125,30(R0) : 2702 |
| | | | MOVB INDEX_AREA_NUMBER,39(R0) : 2703 |
| | | | PUSHAL #0 : 2704 |
| | | | CALLS #1,INSERT_IN_ORDER : 2706 |
| | | | CALLS #0,MAKE_SCRATCH : 2710 |
| | | | MOVL DEF_SCRATCH,R0 : 2712 |
| | | | MOVL IDATA+132,26(R0) : 2719 |
| | | | MOVB #127,30(R0) : 2720 |
| | | | MOVL IDATA,39(R0) : 2721 |
| | | | PUSHAL #0 : 2723 |
| | | | CALLS #1,INSERT_IN_ORDER : 2727 |
| | | | BBS #0,BDATA+T9,90\$: 2731 |
| | | | CALLS #0,MAKE_SCRATCH : 2733 |
| | | | MOVL DEF_SCRATCH,R0 : 2740 |
| | | | MOVL IDATA+132,26(R0) : 2741 |
| | | | MOVB #-123,30(R0) : 2742 |
| | | | MOVL IDATA+216,39(R0) : 2744 |
| | | | PUSHAL #0 : 2752 |
| | | | CALLS #1,INSERT_IN_ORDER : 2752 |
| | | | BRB 95\$: 2752 |
| | | | CLRL R3 : 2752 |

| Generated Code | | | | |
|----------------|----|-----------|---|--------|
| 00000000G | EF | 53 | DO 006AF 91%: MOVL R3,SEGMENT_NUMBER | |
| 00V00000000G | EF | 00 | DO 006B6 MOVL SEGMENT_NUMBER,R0 | : 2756 |
| 00000000G | EF | 00 | E1 006BD BBC #0,SEGMENT_WANTED[R0],94% | |
| 1A | A0 | 00000084G | FB 006C6 CALLS #0,MAKE_SCRATCH | : 2760 |
| 1E | A0 | 85 | DO 006CD MOVL DEF_SCRATCH,R0 | : 2762 |
| 27 | A0 | 00000000G | DO 006D4 MOVL IDATA+132,26(R0) | : 2769 |
| 1F | A0 | 00000000G | 90 006DC MOVB #-123,30(R0) | : 2770 |
| | | 8F | DO 006E1 MOVL SEGMENT_NUMBER,R4 | : 2771 |
| | | EF | DO 006E8 MOVL SEGMENT_LENGTH[R4],39(R0) | |
| | | EF | DO 006F1 MOVL SEGMENT_NUMBER,31(R0) | : 2772 |
| | | 8F | DF 006F9 PUSHAL #0 | : 2774 |
| 00000000G | EF | 01 | FB 006FF CALLS #1,INSERT_IN_ORDER | |
| A5 | 53 | 07 | F3 00706 94%: AOBLEQ #7,R3,91% | |
| 00000000G | EF | 00 | FB 0070A 95%: CALLS #0,MAKE_SCRATCH | : 2782 |
| 1A | A0 | 00000084G | DO 00711 MOVL DEF_SCRATCH,R0 | : 2784 |
| 1E | A0 | 80 | DO 00718 MOVL IDATA+132,26(R0) | : 2791 |
| 27 | A0 | | 90 00720 MOVB #-128,30(R0) | : 2792 |
| | | 55 | DO 00725 MOVL INDEX_AREA_NUMBER,39(R0) | : 2793 |
| | | 8F | DF 00729 PUSHAL #0 | : 2795 |
| 00000000G | EF | 01 | FB 0072F CALLS #1,INSERT_IN_ORDER | |
| 00V00000006G | EF | 00 | E1 00736 BBC #0,BDATA+8,100% | : 2802 |
| 00000000G | EF | 00 | FB 0073E CALLS #0,MAKE_SCRATCH | : 2806 |
| | | 55 | DO 00745 MOVL DEF_SCRATCH,R5 | : 2808 |
| | | 11 | A5 9F 0074C PUSHAB 17(R5) | : 2812 |
| | | 00000018G | EF 9F 0074F PUSHAB SDATA+24 | |
| 00000000G | EF | 02 | FB 00755 CALLS #2,LIB\$SCOPY_DXDX | |
| | | 00000018G | EF 9F 0075C PUSHAB SDATA+24 | : 2813 |
| 00000000G | EF | 01 | FB 00762 CALLS #1,STR\$FREE1_DX | |
| 1A | A5 | 00000084G | EF DO 00769 MOVL IDATA+132,26(R5) | : 2815 |
| 1E | A5 | 81 | 8F 90 00771 MOVB #-127,30(R5) | : 2816 |
| | | 00000000 | 8F DF 00776 PUSHAL #0 | : 2818 |
| 00000000G | EF | 01 | FB 0077C CALLS #1,INSERT_IN_ORDER | |
| | | 00V | 11 00783 BRB 103% | |
| | | 00000000 | 8F DF 00785 100%: PUSHAL #0 | : 2828 |
| | | 81 | 8F 9F 0078B PUSHAB #-127 | |
| | | 00000084G | EF 9F 0078E PUSHAB IDATA+132 | |
| | | 0B | 8F 9F 00794 PUSHAB #11 | |
| | | 01 | 8F 9F 00797 PUSHAB #1 | |
| 00000000G | EF | 05 | FB 0079A CALLS #5,FIND_OBJECT | |
| 00V | 50 | E9 | 007A1 BLBC R0,103% | |
| 00000000G | EF | 00 | FB 007A4 CALLS #0,DELETE_CURRENT | : 2830 |
| | | 00000084G | EF D5 007AB 103%: TSTL IDATA+132 | : 2834 |
| | | 00V | 12 007B1 BNEQ 107% | |
| 00V00000033G | EF | 00 | E1 007B3 BBC #0,VDATA+51,107% | |
| 00000000G | EF | 00 | FB 007BB CALLS #0,MAKE_SCRATCH | : 2842 |
| | | 50 | DO 007C2 MOVL DEF_SCRATCH,R0 | : 2844 |
| 1A | A0 | 00000084G | EF DO 007C9 MOVL IDATA+132,26(R0) | : 2851 |
| 1E | A0 | 84 | 8F 90 007D1 MOVB #-124,30(R0) | : 2852 |
| 27 | A0 | 000000F8G | EF DO 007D6 MOVL IDATA+248,39(R0) | : 2853 |
| | | 00000000 | 8F DF 007DE PUSHAL #0 | : 2855 |
| 00000000G | EF | 01 | FB 007E4 CALLS #1,INSERT_IN_ORDER | |
| 00V00000013G | EF | 00 | E0 007EB 107%: BBS #0,BDATA+T9,T10% | : 2861 |
| 00000000G | EF | 00 | FB 007F3 CALLS #0,MAKE_SCRATCH | : 2865 |
| | | 50 | DO 007FA MOVL DEF_SCRATCH,R0 | : 2867 |
| 1A | A0 | 00000084G | EF DO 00801 MOVL IDATA+132,26(R0) | : 2874 |
| 1E | A0 | 86 | 8F 90 00809 MOVB #-122,30(R0) | : 2875 |
| 27 | A0 | 000000CCG | EF DO 0080E MOVL IDATA+204,39(R0) | : 2876 |

| | | | | | | | | | |
|------------------|-----|-----------|---------------|----|-------|--------|---------------------------------|------------------------------|--------|
| 00000000G | EF | 00000000 | 8F | DF | 00816 | PUSHAL | #0 | : 2878 | |
| | | | 01 | FB | 0081C | CALLS | #1, INSERT_IN_ORDER | | |
| | | | 00V | 11 | 00823 | BRB | 115\$ | | |
| | | | 55 | D4 | 00825 | 110\$: | | : 2886 | |
| 00000000G | EF | | 55 | D0 | 00827 | 111\$: | | | |
| | 50 | 00000000G | EF | D0 | 0082E | MOVL | R5, SEGMENT_NUMBER | : 2890 | |
| 00V00000000GEF40 | | | 00 | E1 | 00835 | MOVL | SEGMENT_NUMBER, R0 | | |
| 00000000G | EF | | 00 | FB | 0083E | BBC | #0, SEGMENT_WANTED[R0], 114\$ | : 2894 | |
| | 50 | 00000000G | EF | D0 | 00845 | CALLS | #0, MAKE_SCRATCH | : 2896 | |
| | 1A | A0 | EF | D0 | 0084C | MOVL | DEF_SCRATCH, R0 | : 2903 | |
| | 1E | A0 | 86 | 8F | 90 | 00854 | MOVL | IDATA+132, 26(R0) | : 2904 |
| | 53 | 00000000G | EF | D0 | 00859 | MOVB | #-122, 30(R0) | : 2905 | |
| | 27 | A0 | 00000000GEF43 | D0 | 00860 | MOVL | SEGMENT_NUMBER, R3 | | |
| | 1F | A0 | 00000000G | EF | D0 | 00869 | MOVL | SEGMENT_POSITION[R3], 39(R0) | : 2906 |
| | | | 00000000 | 8F | DF | 00871 | MOVL | SEGMENT_NUMBER, 31(R0) | : 2908 |
| A5 00000000G | EF | | 01 | FB | 00877 | PUSHAL | #0 | | |
| | 55 | | 07 | F3 | 0087E | CALLS | #1, INSERT_IN_ORDER | | |
| 00000000G | EF | | 00 | FB | 00882 | AOBLEQ | #7, R5, 111\$ | | |
| | 50 | 00000000G | EF | D0 | 00889 | 115\$: | | : 2919 | |
| | 1A | A0 | 00000084G | EF | D0 | 00890 | CALLS | #0, MAKE_SCRATCH | : 2921 |
| | 1E | A0 | 87 | 8F | 90 | 00898 | MOVL | DEF_SCRATCH, R0 | : 2925 |
| | 23 | A0 | 000000DCG | EF | D0 | 0089D | MOVL | IDATA+132, 26(R0) | : 2926 |
| | 1F | A0 | | 07 | D0 | 008A5 | MOVB | #-121, 30(R0) | : 2927 |
| | | | 00000000 | 8F | DF | 008A9 | MOVL | IDATA+220, 35(R0) | : 2932 |
| 00000000G | EF | | 01 | FB | 008AF | MOVL | #7, 31(R0) | : 2934 | |
| | | 00000084G | EF | D5 | 008B6 | PUSHAL | #0 | | |
| | | | 00V | 12 | 008BC | CALLS | #1, INSERT_IN_ORDER | : 2942 | |
| 00000000G | EF | | 00 | FB | 008BE | TSTL | IDATA+132 | | |
| 00V000000014G | EF | | 00 | E1 | 008C5 | BNEQ | 124\$ | : 2946 | |
| 00000000G | EF | | 00 | FB | 008CD | CALLS | #0, ASK_GLOBAL_WANTED | : 2951 | |
| | 50 | 00000000G | EF | D0 | 008D4 | BBC | #0, BDATA+20, 120\$ | : 2955 | |
| | 19 | A0 | 08 | 90 | 008DB | CALLS | #0, MAKE_SCRATCH | : 2957 | |
| | 1E | A0 | 55 | 8F | 90 | 008DF | MOVL | DEF_SCRATCH, R0 | : 2961 |
| | 27 | A0 | 000000B8G | EF | D0 | 008E4 | MOVB | #8, 25(R0) | : 2962 |
| | | | 00000000 | 8F | DF | 008EC | MOVB | #85, 30(R0) | : 2963 |
| 00000000G | EF | | 01 | FB | 008F2 | MOVL | IDATA+184, 39(R0) | : 2965 | |
| | | | 00V | 11 | 008F9 | PUSHAL | #0 | | |
| | | 00000000 | 8F | DF | 008FB | CALLS | #1, INSERT_IN_ORDER | | |
| | | 55 | 8F | 9F | 00901 | 120\$: | | : 2975 | |
| | | 00000000 | 8F | DF | 00904 | BRB | 124\$ | | |
| | | 08 | 8F | 9F | 0090A | PUSHAL | #0 | | |
| | | 01 | 8F | 9F | 0090D | PUSHAB | #8 | | |
| 00000000G | EF | | 05 | FB | 00910 | PUSHAB | #1 | | |
| | 00V | | 50 | E9 | 00917 | CALLS | #5, FIND_OBJECT | | |
| 00000000G | EF | | 00 | FB | 0091A | BLBC | R0, 124\$ | | |
| 55 | 52 | | 01 | C1 | 00921 | CALLS | #0, DELETE_CURRENT | : 2977 | |
| 03 00000000G | EF | | 00 | E1 | 00925 | ADDL3 | #1, CHOSEN_DEPTH, CHOSEN_DEPTH2 | : 2986 | |
| | | | 0000V | 31 | 0092D | BBC | #0, AUTO_TONE...+3 | : 2988 | |
| | | 00000000G | EF | 9F | 00930 | BRW | 127\$ | | |
| | | | 02 | DD | 00936 | PUSHAB | CRLF | : 2992 | |
| | | 00000000G | EF | 9F | 00938 | PUSHL | #2 | | |
| 00000000G | EF | | 03 | FB | 0093E | PUSHAB | PASS\$V OUTPUT | | |
| | | 00000000G | EF | 9F | 00945 | CALLS | #3, PASS\$WRITE_STRING | | |
| | | | 04 | DD | 0094B | PUSHAB | SHIFT | | |
| | | 00000000G | EF | 9F | 0094D | PUSHL | #4 | | |
| 00000000G | EF | | 03 | FB | 00953 | PUSHAB | PASS\$V OUTPUT | | |
| | | FFFFE549 | EF | 9F | 0095A | CALLS | #3, PASS\$WRITE_STRING | | |
| | | | | | | PUSHAB | C.AAJ | | |

| | | | | | | | | | |
|-----------|----|-----------|----|-------|-------|--------|----------------------|--|--------|
| 00000000G | EF | 00000000G | 10 | DD | 00960 | PUSHL | #16 | | |
| | | | EF | 9F | 00962 | PUSHAB | PASSFV OUTPUT | | |
| | | | 03 | FB | 00968 | CALLS | #3,PASSWRITE_STRING | | |
| | | | 03 | DD | 0096F | PUSHL | #3 | | |
| | | 00000084G | EF | DD | 00971 | PUSHL | IDATA+132 | | |
| 00000000G | EF | 00000000G | EF | 9F | 00977 | PUSHAB | PASSFV OUTPUT | | |
| | | | 03 | FB | 0097D | CALLS | #3,PASSWRITE_INTEGER | | |
| | | FFFFE52F | EF | 9F | 00984 | PUSHAB | C,AAK | | |
| | | | 1E | DD | 0098A | PUSHL | #30 | | |
| 00000000G | EF | 00000000G | EF | 9F | 0098C | PUSHAB | PASSFV OUTPUT | | |
| | | | 03 | FB | 00992 | CALLS | #3,PASSWRITE_STRING | | |
| | | 00000000G | EF | 9F | 00999 | PUSHAB | CRLF_SHIFT | | |
| | | | 06 | DD | 0099F | PUSHL | #6 | | |
| 00000000G | EF | 00000000G | EF | 9F | 009A1 | PUSHAB | PASSFV OUTPUT | | |
| | | | 03 | FB | 009A7 | CALLS | #3,PASSWRITE_STRING | | |
| | | FFFFE525 | EF | 9F | 009AE | PUSHAB | C,AAL | | |
| | | | 05 | DD | 009B4 | PUSHL | #5 | | |
| 00000000G | EF | 00000000G | EF | 9F | 009B6 | PUSHAB | PASSFV OUTPUT | | |
| | AD | | 03 | FB | 009BC | CALLS | #3,PASSWRITE_STRING | | |
| | | FC | 52 | DD | 009C3 | MOVL | CHOSEN_DEPTH,-4(FP) | | |
| 00000000G | EF | | AD | 9F | 009C7 | PUSHAB | -4(FP) | | |
| | | | 01 | FB | 009CA | CALLS | #1,NUM_LEN | | |
| | | | 50 | DD | 009D1 | PUSHL | R0 | | |
| | | | 52 | DD | 009D3 | PUSHL | CHOSEN_DEPTH | | |
| 00000000G | EF | 00000000G | EF | 9F | 009D5 | PUSHAB | PASSFV OUTPUT | | |
| | | | 03 | FB | 009DB | CALLS | #3,PASSWRITE_INTEGER | | |
| | | FFFFE4F9 | EF | 9F | 009E2 | PUSHAB | C,AAM | | |
| | | | 18 | DD | 009E8 | PUSHL | #24 | | |
| 00000000G | EF | 00000000G | EF | 9F | 009EA | PUSHAB | PASSFV OUTPUT | | |
| | AD | | 03 | FB | 009F0 | CALLS | #3,PASSWRITE_STRING | | |
| | | FC | 55 | DD | 009F7 | MOVL | CHOSEN_DEPTH2,-4(FP) | | |
| 00000000G | EF | | AD | 9F | 009FB | PUSHAB | -4(FP) | | |
| | | | 01 | FB | 009FE | CALLS | #1,NUM_LEN | | |
| | | | 50 | DD | 00A05 | PUSHL | R0 | | |
| | | | 55 | DD | 00A07 | PUSHL | CHOSEN_DEPTH2 | | |
| 00000000G | EF | 00000000G | EF | 9F | 00A09 | PUSHAB | PASSFV OUTPUT | | |
| | | | 03 | FB | 00A0F | CALLS | #3,PASSWRITE_INTEGER | | |
| | | FFFFE4DD | EF | 9F | 00A16 | PUSHAB | C,AAN | | |
| | | | 0E | DD | 00A1C | PUSHL | #14 | | |
| 00000000G | EF | 00000000G | EF | 9F | 00A1E | PUSHAB | PASSFV OUTPUT | | |
| | | | 03 | FB | 00A24 | CALLS | #3,PASSWRITE_STRING | | |
| | | 00000000G | EF | 9F | 00A2B | PUSHAB | PASSFV OUTPUT | | |
| 00000000G | EF | | 01 | FB | 00A31 | CALLS | #1,PASSWRITELN2 | | |
| | | 0000001F | 8F | DF | 00A38 | PUSHAL | #31 | | : 3001 |
| 00000000G | EF | | 01 | FB | 00A3E | CALLS | #1,QUERY | | : 3005 |
| | | | 04 | 00A45 | 1278: | RET | | | |

; Routine Size: 2630 bytes. Routine Base: \$CODE + 0127F

| | | | | | | | | |
|-----------|----|-----------|----|------|-------|---------------|----------------------|--------|
| 00000000G | EF | 00000003 | 00 | 0000 | 00000 | LINK_RESULTS: | | : 3052 |
| | | | 8F | FB | 00002 | .WORD | *M<> | |
| 00000000G | EF | | 01 | DF | 00009 | CALLS | #0,EDF\$RESET_SCROLL | : 3059 |
| | | 00000000G | EF | 94 | 00016 | PUSHAL | #3 | : 3060 |
| | | 00000000G | EF | 94 | 0001C | CALLS | #1,CLEAR | |
| 00000000G | EF | | 01 | 90 | 00022 | CLRB | VISIBLE_QUESTION | : 3061 |
| | | | | | | CLRB | WAIT_HELP | : 3062 |
| | | | | | | MOVB | #1,TAKE_DEFAULTS | : 3063 |

| | | | | | | |
|--------------|-----|----|-------|-------|----------------|--------|
| 00000084G | EF | D5 | 00029 | TSTL | 1DATA+132 | : 3068 |
| | 00V | 12 | 0002F | BNEQ | 2\$ | |
| 0D02 CF | 00 | FB | 00031 | CALLS | #0,NON_KEY_DEF | : 3070 |
| 127F CF | 00 | FB | 00036 | CALLS | #0,APPEND_DEF | : 3075 |
| 00000000G EF | 01 | 90 | 00038 | MOVB | #1,LINKED | : 3077 |
| | | 04 | 00042 | RET | | : 3079 |

; Routine Size: 67 bytes, Routine Base: \$CODE + 01CC5

| | | | | | | |
|-----------|-----|-----------|-------|-------------|-------------------------------------|--------------------------------|
| | | | 00000 | MERGE_AREA: | | : 3132 |
| | | OFFC | 00000 | .WORD | *M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> | |
| | 5E | 08 | C2 | 00002 | SUBL2 | #8,SP |
| | 52 | 04 | BC | D0 | 00005 | MOVL @4(R12),CURKEY |
| FC | AD | 08 | BC | D0 | 00009 | MOVL @8(R12),MAXKEY |
| | 54 | 0C | BC | D0 | 0000E | MOVL @12(R12),SRCDATA |
| | 55 | 10 | BC | D0 | 00012 | MOVL @16(R12),DSTDATA |
| | 56 | 14 | BC | D0 | 00016 | MOVL @20(R12),SRCIDX |
| | 5C | 18 | BC | D0 | 0001A | MOVL @24(R12),DSTIDX |
| | 57 | | 03 | D0 | 0001E | MOVL #3,SOURCE_DATA_BUCKET |
| | | | 58 | D4 | 00021 | CLRL SOURCE_DATA_ALLOC |
| | | | 59 | D4 | 00023 | CLRL SOURCE_DATA_EXT |
| | 5A | | 03 | D0 | 00025 | MOVL #3,SOURCE_INDEX_BUCKET |
| | | | 5B | D4 | 00028 | CLRL SOURCE_INDEX_ALLOC |
| | | | 53 | D4 | 0002A | CLRL SOURCE_INDEX_EXT |
| | | 00000000 | 8F | DF | 0002C | PUSHAL #0 |
| | | 1D | 8F | 9F | 00032 | PUSHAB #29 |
| F8 | AD | | 54 | D0 | 00035 | MOVL SRCDATA,-8(FP) |
| | | F8 | AD | 9F | 00039 | PUSHAB -8(FP) |
| | | 05 | 8F | 9F | 0003C | PUSHAB #5 |
| | | 01 | 8F | 9F | 0003F | PUSHAB #1 |
| 00000000G | EF | 05 | FB | 00042 | CALLS #5,FIND_OBJECT | |
| | 00V | 50 | E9 | 00049 | BLBC R0,2\$ | |
| | 50 | 00000000G | EF | D0 | 0004C | MOVL DEF_CURRENT,R0 |
| | 57 | 27 | A0 | D0 | 00053 | MOVL 39(R0),SOURCE_DATA_BUCKET |
| | | 00000000 | 8F | DF | 00057 | PUSHAL #0 |
| | | 18 | 8F | 9F | 0005D | PUSHAB #27 |
| F8 | AD | | 54 | D0 | 00060 | MOVL SRCDATA,-8(FP) |
| | | F8 | AD | 9F | 00064 | PUSHAB -8(FP) |
| | | 05 | 8F | 9F | 00067 | PUSHAB #5 |
| | | 01 | 8F | 9F | 0006A | PUSHAB #1 |
| 00000000G | EF | 05 | FB | 0006D | CALLS #5,FIND_OBJECT | |
| | 00V | 50 | E9 | 00074 | BLBC R0,4\$ | |
| | 50 | 00000000G | EF | D0 | 00077 | MOVL DEF_CURRENT,R0 |
| | 58 | 27 | A0 | D0 | 0007E | MOVL 39(R0),SOURCE_DATA_ALLOC |
| | | 00000000 | 8F | DF | 00082 | PUSHAL #0 |
| | | 20 | 8F | 9F | 00088 | PUSHAB #32 |
| F8 | AD | | 54 | D0 | 0008B | MOVL SRCDATA,-8(FP) |
| | | F8 | AD | 9F | 0008F | PUSHAB -8(FP) |
| | | 05 | 8F | 9F | 00092 | PUSHAB #5 |
| | | 01 | 8F | 9F | 00095 | PUSHAB #1 |
| 00000000G | EF | 05 | FB | 00098 | CALLS #5,FIND_OBJECT | |
| | 00V | 50 | E9 | 0009F | BLBC R0,6\$ | |
| | 50 | 00000000G | EF | D0 | 000A2 | MOVL DEF_CURRENT,R0 |
| | 59 | 27 | A0 | D0 | 000A9 | MOVL 39(R0),SOURCE_DATA_EXT |
| | | 00000000 | 8F | DF | 000AD | PUSHAL #0 |
| | | 1D | 8F | 9F | 000B3 | PUSHAB #29 |
| F8 | AD | | 56 | D0 | 000B6 | MOVL SRCIDX,-8(FP) |

Generated Code

| | | | | | | | | |
|-----------|-----------|-----------|-----|----|-------|--------|----------------------------|--------|
| | | F8 | AD | 9F | 000BA | PUSHAB | -8(FP) | |
| | | 05 | 8F | 9F | 000BD | PUSHAB | #5 | |
| | | 01 | 8F | 9F | 000C0 | PUSHAB | #1 | |
| 00000000G | EF | | 05 | FB | 000C3 | CALLS | #5,FIND_OBJECT | |
| | 00V | | 50 | E9 | 000CA | BLBC | R0,8\$ | |
| | 50 | 00000000G | EF | D0 | 000CD | MOVL | DEF CURRENT,R0 | : 3177 |
| | 5A | 27 | A0 | D0 | 000D4 | MOVL | 39(R0),SOURCE_INDEX_BUCKET | |
| | | 00000000 | 8F | DF | 000D8 | PUSHAL | #0 | : 3179 |
| | | 1B | 8F | 9F | 000DE | PUSHAB | #27 | |
| F8 | AD | | 56 | D0 | 000E1 | MOVL | SRCIDX,-8(FP) | |
| | | F8 | AD | 9F | 000E5 | PUSHAB | -8(FP) | |
| | | 05 | 8F | 9F | 000E8 | PUSHAB | #5 | |
| | | 01 | 8F | 9F | 000EB | PUSHAB | #1 | |
| 00000000G | EF | | 05 | FB | 000EE | CALLS | #5,FIND_OBJECT | |
| | 00V | | 50 | E9 | 000F5 | BLBC | R0,10\$ | |
| | 50 | 00000000G | EF | D0 | 000F8 | MOVL | DEF CURRENT,R0 | : 3181 |
| | 5B | 27 | A0 | D0 | 000FF | MOVL | 39(R0),SOURCE_INDEX_ALLOC | |
| | | 00000000 | 8F | DF | 00103 | PUSHAL | #0 | : 3183 |
| | | 20 | 8F | 9F | 00109 | PUSHAB | #32 | |
| F8 | AD | | 56 | D0 | 0010C | MOVL | SRCIDX,-8(FP) | |
| | | F8 | AD | 9F | 00110 | PUSHAB | -8(FP) | |
| | | 05 | 8F | 9F | 00113 | PUSHAB | #5 | |
| | | 01 | 8F | 9F | 00116 | PUSHAB | #1 | |
| 00000000G | EF | | 05 | FB | 00119 | CALLS | #5,FIND_OBJECT | |
| | 00V | | 50 | E9 | 00120 | BLBC | R0,12\$ | |
| | 50 | 00000000G | EF | D0 | 00123 | MOVL | DEF CURRENT,R0 | : 3185 |
| | 53 | 27 | A0 | D0 | 0012A | MOVL | 39(R0),SOURCE_INDEX_EXT | |
| | 55 | | 54 | D1 | 0012E | CMPL | SRCDATA,DSTDATA | : 3191 |
| | | | 00V | 12 | 00131 | BNEQ | 14\$ | |
| | | | 57 | D4 | 00133 | CLRL | SOURCE_DATA_BUCKET | : 3195 |
| | | | 58 | D4 | 00135 | CLRL | SOURCE_DATA_ALLOC | : 3196 |
| | | | 59 | D4 | 00137 | CLRL | SOURCE_DATA_EXT | : 3197 |
| | | | 56 | D1 | 00139 | CMPL | SRCIDX,DSTIDX | : 3201 |
| | 5C | | 00V | 12 | 0013C | BNEQ | 16\$ | |
| | | | 5A | D4 | 0013E | CLRL | SOURCE_INDEX_BUCKET | : 3205 |
| | | | 5B | D4 | 00140 | CLRL | SOURCE_INDEX_ALLOC | : 3206 |
| | | | 53 | D4 | 00142 | CLRL | SOURCE_INDEX_EXT | : 3207 |
| | | 00000000 | 8F | DF | 00144 | PUSHAL | #0 | : 3215 |
| | | 1D | 8F | 9F | 0014A | PUSHAB | #29 | |
| F8 | AD | | 55 | D0 | 0014D | MOVL | DSTDATA,-8(FP) | |
| | | F8 | AD | 9F | 00151 | PUSHAB | -8(FP) | |
| | | 05 | 8F | 9F | 00154 | PUSHAB | #5 | |
| | | 01 | 8F | 9F | 00157 | PUSHAB | #1 | |
| 00000000G | EF | | 05 | FB | 0015A | CALLS | #5,FIND_OBJECT | |
| | 00V | | 50 | E9 | 00161 | BLBC | R0,20\$ | |
| 50 | 00000000G | | 27 | C1 | 00164 | ADDL3 | #39,DEF CURRENT,R0 | : 3217 |
| | 60 | | 57 | D1 | 0016C | CMPL | SOURCE_DATA_BUCKET,(R0) | |
| | | | 00V | 15 | 0016F | BLEQ | 20\$ | |
| | 60 | | 57 | D0 | 00171 | MOVL | SOURCE_DATA_BUCKET,(R0) | : 3219 |
| | | 00000000 | 8F | DF | 00174 | PUSHAL | #0 | : 3221 |
| | | 1B | 8F | 9F | 0017A | PUSHAB | #27 | |
| F8 | AD | | 55 | D0 | 0017D | MOVL | DSTDATA,-8(FP) | |
| | | F8 | AD | 9F | 00181 | PUSHAB | -8(FP) | |
| | | 05 | 8F | 9F | 00184 | PUSHAB | #5 | |
| | | 01 | 8F | 9F | 00187 | PUSHAB | #1 | |
| 00000000G | EF | | 05 | FB | 0018A | CALLS | #5,FIND_OBJECT | |
| | 00V | | 50 | E9 | 00191 | BLBC | R0,22\$ | |

| Generated Code | | | |
|----------------|----------|----|----------------|
| 50 00000000G | EF | 27 | C1 00194 |
| | 60 | 58 | C0 0019C |
| | 00000000 | 8F | DF 0019F 22\$: |
| | 20 | 8F | 9F 001A5 |
| F8 | AD | 55 | D0 001A8 |
| | F8 | AD | 9F 001AC |
| | 05 | 8F | 9F 001AF |
| | 01 | 8F | 9F 001B2 |
| 00000000G | EF | 05 | FB 001B5 |
| | 00V | 50 | E9 001BC |
| 50 00000000G | EF | 27 | C1 001BF |
| | 60 | 59 | C0 001C7 |
| | 00000000 | 8F | DF 001CA 24\$: |
| | 1D | 8F | 9F 001D0 |
| F8 | AD | 5C | D0 001D3 |
| | F8 | AD | 9F 001D7 |
| | 05 | 8F | 9F 001DA |
| | 01 | 8F | 9F 001DD |
| 00000000G | EF | 05 | FB 001E0 |
| | 00V | 50 | E9 001E7 |
| 50 00000000G | EF | 27 | C1 001EA |
| | 60 | 5A | D1 001F2 |
| | 00V | 15 | 001F5 |
| | 60 | 5A | D0 001F7 |
| | 00000000 | 8F | DF 001FA 28\$: |
| | 1B | 8F | 9F 00200 |
| F8 | AD | 5C | D0 00203 |
| | F8 | AD | 9F 00207 |
| | 05 | 8F | 9F 0020A |
| | 01 | 8F | 9F 0020D |
| 00000000G | EF | 05 | FB 00210 |
| | 00V | 50 | E9 00217 |
| 50 00000000G | EF | 27 | C1 0021A |
| | 60 | 5B | C0 00222 |
| | 00000000 | 8F | DF 00225 30\$: |
| | 20 | 8F | 9F 0022B |
| F8 | AD | 5C | D0 0022E |
| | F8 | AD | 9F 00232 |
| | 05 | 8F | 9F 00235 |
| | 01 | 8F | 9F 00238 |
| 00000000G | EF | 05 | FB 0023B |
| | 00V | 50 | E9 00242 |
| 50 00000000G | EF | 27 | C1 00245 |
| | 60 | 53 | C0 0024D |
| | 53 | AD | D0 00250 32\$: |
| | 53 | D1 | 00254 |
| | 00V | 15 | 00257 |
| | 0000V | 31 | 00259 |
| | 52 | D6 | 0025C 33\$: |
| | 52 | D0 | 0025E 34\$: |
| | 00000000 | 8F | DF 00261 |
| | 78 | 8F | 9F 00267 |
| | 5B | D0 | 0026A |
| FC | AD | AD | 9F 0026E |
| | FC | 8F | 9F 00271 |
| | 08 | 8F | 9F 00274 |
| | 01 | 05 | FB 00277 |
| 00000000G | EF | 05 | FB 00277 |

| | | |
|--------|--------------------------|--------|
| ADDL3 | #39,DEF_CURRENT,R0 | : 3223 |
| ADDL2 | SOURCE_DATA_ALLOC,(R0) | |
| PUSHAL | #0 | : 3225 |
| PUSHAB | #32 | |
| MOVL | DSTDATA,-8(FP) | |
| PUSHAB | -8(FP) | |
| PUSHAB | #5 | |
| PUSHAB | #1 | |
| CALLS | #5,FIND_OBJECT | |
| BLBC | R0,24\$ | |
| ADDL3 | #39,DEF_CURRENT,R0 | : 3227 |
| ADDL2 | SOURCE_DATA_EXT,(R0) | |
| PUSHAL | #0 | : 3229 |
| PUSHAB | #29 | |
| MOVL | DSTIDX,-8(FP) | |
| PUSHAB | -8(FP) | |
| PUSHAB | #5 | |
| PUSHAB | #1 | |
| CALLS | #5,FIND_OBJECT | |
| BLBC | R0,28\$ | |
| ADDL3 | #39,DEF_CURRENT,R0 | : 3231 |
| CMPL | SOURCE_INDEX_BUCKET,(R0) | |
| BLEQ | 28\$ | |
| MOVL | SOURCE_INDEX_BUCKET,(R0) | : 3233 |
| PUSHAL | #0 | : 3235 |
| PUSHAB | #27 | |
| MOVL | DSTIDX,-8(FP) | |
| PUSHAB | -8(FP) | |
| PUSHAB | #5 | |
| PUSHAB | #1 | |
| CALLS | #5,FIND_OBJECT | |
| BLBC | R0,30\$ | |
| ADDL3 | #39,DEF_CURRENT,R0 | : 3237 |
| ADDL2 | SOURCE_INDEX_ALLOC,(R0) | |
| PUSHAL | #0 | : 3239 |
| PUSHAB | #32 | |
| MOVL | DSTIDX,-8(FP) | |
| PUSHAB | -8(FP) | |
| PUSHAB | #5 | |
| PUSHAB | #1 | |
| CALLS | #5,FIND_OBJECT | |
| BLBC | R0,32\$ | |
| ADDL3 | #39,DEF_CURRENT,R0 | : 3241 |
| ADDL2 | SOURCE_INDEX_EXT,(R0) | |
| MOVL | MAXKEY,R3 | : 3243 |
| CMPL | R2,R3 | |
| BLEQ | 34\$ | |
| BRW | 41\$ | |
| INCL | R2 | |
| MOVL | R2,KEYNUM | |
| PUSHAL | #0 | : 3250 |
| PUSHAB | #120 | |
| MOVL | KEYNUM,-4(FP) | |
| PUSHAB | -4(FP) | |
| PUSHAB | #11 | |
| PUSHAB | #1 | |
| CALLS | #5,FIND_OBJECT | |

Generated Code

```
00V 50 E9 0027E BLBC R0,36$
27 50 00000000G EF D0 00281 MOVL DEF_CURRENT,R0 ; 3252
AO 8F DF 00288 MOVL DSTDATA,39(R0)
00000000 8F 9F 00292 36$: PUSHAL #0 ; 3254
7D 8F 9F 00295 MOVL KEYNUM,-4(FP)
FC AD FC AD 9F 00299 PUSHAB -4(FP)
OB 8F 9F 0029C PUSHAB #11
01 8F 9F 0029F PUSHAB #1
00000000G EF 05 FB 002A2 CALLS #5,FIND_OBJECT
00V 50 E9 002A9 BLBC R0,38$
27 50 00000000G EF D0 002AC MOVL DEF_CURRENT,R0 ; 3256
AO 5C D0 002B3 MOVL DSTIDX,39(R0)
00000000 8F DF 002B7 38$: PUSHAL #0 ; 3258
80 8F 9F 002BD PUSHAB #-128
FC AD 5B D0 002C0 MOVL KEYNUM,-4(FP)
FC AD 9F 002C4 PUSHAB -4(FP)
OB 8F 9F 002C7 PUSHAB #11
01 8F 9F 002CA PUSHAB #1
00000000G EF 05 FB 002CD CALLS #5,FIND_OBJECT
00V 50 E9 002D4 BLBC R0,40$
27 50 00000000G EF D0 002D7 MOVL DEF_CURRENT,R0 ; 3260
AO 5C D0 002DE MOVL DSTIDX,39(R0)
53 52 D1 002E2 40$: CMPL R2,R3
03 18 002E5 BGEQ +3
FF72 31 002E7 BRW 33$
55 54 D1 002EA 41$: CMPL SRCDATA,DSTDATA ; 3267
00V 13 002ED BEQL 43$
FC AD 54 D0 002EF MOVL SRCDATA,-4(FP) ; 3269
FC AD 9F 002F3 PUSHAB -4(FP)
05 8F 9F 002F6 PUSHAB #5
00000000G EF 02 FB 002F9 CALLS #2,DELETE_PRIMARY_SECTION ; 3271
5C 56 D1 00300 43$: CMPL SRCIDX,DSTIDX
00V 13 00303 BEQL 45$
FC AD 56 D0 00305 MOVL SRCIDX,-4(FP) ; 3273
FC AD 9F 00309 PUSHAB -4(FP)
05 8F 9F 0030C PUSHAB #5
00000000G EF 02 FB 0030F CALLS #2,DELETE_PRIMARY_SECTION ; 3275
04 00316 45$: RET
```

; Routine Size: 791 bytes, Routine Base: \$CODE + 01D08

```
00000 SHUFFLE_AREAS: ; 3323
0004 00000 .WORD ^M<R2>
5E 01 10 C2 00002 SUBL2 #16,SP
00000000G EF 01 8F 9F 00005 PUSHAB #1 ; 3336
01 00000000G EF FB 00008 CALLS #1,SCAN_DEFINITION ; 3351
00V D1 0000F CMPL HIGH_KEY,#1
04 00000BCG EF 15 00016 BLEQ 5$
00V D1 00018 CMPL IDATA+188,#4
5C 00000000G EF 13 0001F BEQL 5$
52 5C 02 C5 00021 MOVL HIGH_KEY,TEMP_KEY ; 3359
00000003 8F DF 00028 3$: MULL3 #2,TEMP_KEY,TEMP_AREA ; 3366
FC AD 52 01 C1 00032 ADDL3 #1,TEMP_AREA,-4(FP) ; 3368
FC AD 9F 00037 PUSHAB -4(FP)
00000002 8F DF 0003A PUSHAL #2
```


| Generated Code | | | |
|----------------|-----------|-------------|------------------------------|
| F8 | AD | | 52 D0 00040 |
| | | F8 | AD 9F 00044 |
| F4 | AD | | 5C D0 00047 |
| | | F4 | AD 9F 0004B |
| F0 | AD | | 5C D0 0004E |
| | | F0 | AD 9F 00052 |
| 1D08 | CF | | 06 FB 00055 |
| | | | 5C D7 0005A |
| 02 | | | 5C D1 0005C |
| | | | C7 18 0005F |
| 04 | 00 | 000000BCG | EF CF 00061 5\$: |
| | | 0000V | 00069 |
| | | 0000V | 0006B |
| | | 0000V | 0006D |
| | | 0000V | 0006F |
| | | 0000V | 00071 |
| | | 0000V | 31 00073 |
| 01 | 00000000G | EF | D1 00076 6\$: |
| | | 00V | 15 0007D |
| | 00000000 | 8F | DF 0007F |
| | 00000003 | 8F | DF 00085 |
| | 00000000 | 8F | DF 0008B |
| | 00000002 | 8F | DF 00091 |
| | 00000000G | EF | 9F 00097 |
| | 00000001 | 8F | DF 0009D |
| 1D08 | CF | | 06 FB 000A3 |
| | | | 8F DF 000A8 8\$: |
| | 00000001 | 8F | DF 000AE |
| | 00000000 | 8F | DF 000B4 |
| | 00000000 | 8F | DF 000BA |
| | 00000000G | EF | 9F 000C0 |
| | 00000000 | 8F | DF 000C6 |
| 1D08 | CF | | 06 FB 000CC |
| | | 00V | 11 000D1 |
| | 00000000G | EF | D5 000D3 9\$: |
| | | 00V | 15 000D9 |
| | 00000001 | 8F | DF 000DB |
| | 00000003 | 8F | DF 000E1 |
| | 00000001 | 8F | DF 000E7 |
| | 00000002 | 8F | DF 000ED |
| | 00000000G | EF | 9F 000F3 |
| | 00000001 | 8F | DF 000F9 |
| 1D08 | CF | | 06 FB 000FF |
| | | 00V | 11 00104 |
| | 00000002 | 8F | DF 00106 12\$: |
| | 00000003 | 8F | DF 0010C |
| | 00000002 | 8F | DF 00112 |
| | 00000002 | 8F | DF 00118 |
| | 00000000G | EF | 9F 0011E |
| | 00000001 | 8F | DF 00124 |
| 1D08 | CF | | 06 FB 0012A |
| | | 00V | 11 0012F |
| | | 00V | 11 00131 13\$: |
| | | 00V | 11 00133 14\$: |
| | | | 00135 15\$: |
| | | | 00135 16\$: |
| | 00000000G | 52 D4 00135 | |
| | | EF D5 00137 | |
| | | | MOVL TEMP_AREA,-8(FP) |
| | | | PUSHAB -8(FP) |
| | | | MOVL TEMP_KEY,-12(FP) |
| | | | PUSHAB -12(FP) |
| | | | MOVL TEMP_KEY,-16(FP) |
| | | | PUSHAB -16(FP) |
| | | | CALLS #6,MERGE_AREA |
| | | | DECL TEMP_KEY ; 3370 |
| | | | CMPL TEMP_KEY,#2 |
| | | | BGEQ 3\$ |
| | | | CASEL IDATA+188,#0,#4 ; 3376 |
| | | | .DISPL 6\$ |
| | | | .DISPL 9\$ |
| | | | .DISPL 12\$ |
| | | | .DISPL 13\$ |
| | | | .DISPL 14\$ |
| | | | .DISPL 15\$ |
| | | | BRW 15\$ |
| | | | CMPL HIGH_AREA,#1 ; 3382 |
| | | | BLEQ 8\$ |
| | | | PUSHAL #0 ; 3384 |
| | | | PUSHAL #3 |
| | | | PUSHAL #0 |
| | | | PUSHAL #2 |
| | | | PUSHAB HIGH_KEY |
| | | | PUSHAL #1 |
| | | | CALLS #6,MERGE_AREA |
| | | | PUSHAL #0 ; 3386 |
| | | | PUSHAL #1 |
| | | | PUSHAL #0 |
| | | | PUSHAL #0 |
| | | | PUSHAB HIGH_KEY |
| | | | PUSHAL #0 |
| | | | CALLS #6,MERGE_AREA |
| | | | BRB 16\$ |
| | | | TSTL HIGH_KEY ; 3395 |
| | | | BLEQ 16\$ |
| | | | PUSHAL #1 ; 3399 |
| | | | PUSHAL #3 |
| | | | PUSHAL #1 |
| | | | PUSHAL #2 |
| | | | PUSHAB HIGH_KEY |
| | | | PUSHAL #1 |
| | | | CALLS #6,MERGE_AREA |
| | | | BRB 16\$ |
| | | | PUSHAL #2 ; 3407 |
| | | | PUSHAL #3 |
| | | | PUSHAL #2 |
| | | | PUSHAL #2 |
| | | | PUSHAB HIGH_KEY |
| | | | PUSHAL #1 |
| | | | CALLS #6,MERGE_AREA |
| | | | BRB 16\$ |
| | | | BRB 16\$ |
| | | | BRB 16\$ |
| | | | CLRL PROLOG_FOR_KEYS ; 3436 |
| | | | TSTL HIGH_KEY ; 3442 |

| | | | | | | | | | | | |
|----|-----------|----|-----------|-----|-----|-------|-------|--------|---|---|------|
| 50 | 00000000G | 52 | 00000000G | EF | 00V | 15 | 0013D | BLEQ | 21\$ | | |
| 50 | | EF | | 00 | 05 | C7 | 0013F | DIVL3 | #5,HIGH_KEY,PROLOG_FOR_KEYS | : | 3446 |
| 50 | | 50 | | 50 | 00 | 7A | 00147 | EMUL | #0,#0,HIGH_KEY,R0 | : | 3448 |
| | | | | | 05 | 7B | 00150 | EDIV | #5,R0,R0,R0 | | |
| | | | | | 50 | D5 | 00155 | TSTL | R0 | | |
| | | | | | 00V | 18 | 00157 | BGEQ | 18\$ | | |
| | | | 50 | | 05 | C0 | 00159 | ADDL2 | #5,R0 | | |
| | | | | | 50 | D5 | 0015C | TSTL | R0 | | |
| | | | | | 00V | 13 | 0015E | BEQL | 21\$ | | |
| | | | | | 52 | D6 | 00160 | INCL | PROLOG_FOR_KEYS | : | 3450 |
| | | | | | 52 | D6 | 00162 | INCL | PROLOG_FOR_KEYS | : | 3457 |
| | | 50 | 00000000G | EF | 01 | C1 | 00164 | ADDL3 | #1,HIGH_AREA,R0 | : | 3463 |
| | | 5C | | 50 | 07 | C7 | 0016C | DIVL3 | #7,R0,PROLOG_FOR_AREAS | | |
| 50 | | 50 | | 00 | 00 | 7A | 00170 | EMUL | #0,#0,R0,R0 | : | 3465 |
| 50 | | 50 | | 50 | 07 | 7B | 00175 | EDIV | #7,R0,R0,R0 | | |
| | | | | | 50 | D5 | 0017A | TSTL | R0 | | |
| | | | | | 00V | 18 | 0017C | BGEQ | 22\$ | | |
| | | | 50 | | 07 | C0 | 0017E | ADDL2 | #7,R0 | | |
| | | | | | 50 | D5 | 00181 | TSTL | R0 | | |
| | | | | | 00V | 13 | 00183 | BEQL | 24\$ | | |
| | | | | | 5C | D6 | 00185 | INCL | PROLOG_FOR_AREAS | : | 3467 |
| | | | 00000000 | | 8F | DF | 00187 | PUSHAL | #0 | : | 3473 |
| | | | 18 | | 8F | 9F | 0018D | PUSHAB | #27 | | |
| | | | 00000000 | | 8F | DF | 00190 | PUSHAL | #0 | | |
| | | | 05 | | 8F | 9F | 00196 | PUSHAB | #5 | | |
| | | | 01 | | 8F | 9F | 00199 | PUSHAB | #1 | | |
| | | | 00000000G | EF | 05 | FB | 0019C | CALLS | #5,FIND_OBJECT | | |
| | | | | 00V | 50 | E9 | 001A3 | BLBC | R0,26\$ | | |
| | | | 00000059 | | 8F | DF | 001A6 | PUSHAL | #89 | : | 3475 |
| FC | AD | | | 52 | 5C | C1 | 001AC | ADDL3 | PROLOG_FOR_AREAS,PROLOG_FOR_KEYS,-4(FP) | | |
| | | | | | AD | 9F | 001B1 | PUSHAB | -4(FP) | | |
| | | | 00000080G | | EF | 9F | 001B4 | PUSHAB | IDATA+128 | | |
| | | | | | 03 | FB | 001BA | CALLS | #3,MAX_FACTOR | | |
| | | 5C | 00000000G | EF | 27 | C1 | 001C1 | ADDL3 | #39,DEF_CURRENT,R12 | | |
| | | | 00000000G | EF | 50 | C0 | 001C9 | ADDL2 | R0,(R12) | | |
| | | | | 6C | 04 | 001CC | 26\$: | RET | | : | 3480 |

; Routine Size: 461 bytes. Routine Base: \$CODE + 0201F

| | | | | | | | | | | | |
|----|--|--|--------------|-----|-------|-------------|--------|--------------------------|-----------------|---|------|
| | | | | | 00000 | CALC_ARRAY: | | | | : | 3526 |
| | | | | | 01FC | 00000 | .WORD | #M<R2,R3,R4,R5,R6,R7,R8> | | | |
| | | | 00000000G | EF | 9F | 00002 | PUSHAB | SHIFT | | : | 3536 |
| | | | | 04 | DD | 00008 | PUSHL | #4 | | | |
| | | | 00000000G | EF | 9F | 0000A | PUSHAB | PASS\$V_OUTPUT | | | |
| | | | | 03 | FB | 00010 | CALLS | #3,PASS\$WRITE_STRING | | | |
| | | | FFFFDF7F | EF | 9F | 00017 | PUSHAB | C.AAO | | | |
| | | | | 0B | DD | 0001D | PUSHL | #11 | | | |
| | | | 00000000G | EF | 9F | 0001F | PUSHAB | PASS\$V_OUTPUT | | | |
| | | | | 03 | FB | 00025 | CALLS | #3,PASS\$WRITE_STRING | | | |
| | | | 00000000G | EF | 9F | 0002C | PUSHAB | PASS\$V_OUTPUT | | | |
| | | | | 01 | FB | 00032 | CALLS | #1,PASS\$WRITELN2 | | | |
| | | | 00000118G | EF | D5 | 00039 | TSTL | IDATA+280 | | : | 3538 |
| | | | | 00V | 12 | 0003F | BNEQ | 2\$ | | | |
| | | | 00000000G | EF | 02 | D0 | 00041 | MOVL | #2,GRAPH_TYPE | : | 3540 |
| | | | | 00V | 11 | 00048 | BRB | 3\$ | | | |
| | | | 00000000G | EF | 01 | D0 | 0004A | MOVL | #1,GRAPH_TYPE | : | 3544 |
| 04 | | | 00 00000118G | EF | CF | 00051 | 3\$: | CASEL | IDATA+280,#0,#4 | : | 3546 |

| | | | | | | | | | | | | |
|-----------|----|-----------|----|-----------|-------|----|-------|------|--------|-------------------------------------|--------|--|
| 00000000G | EF | FFFFDF36 | EF | 00000014G | 0000V | 31 | 00063 | 48: | .DISPL | 48 | | |
| | | 000000ACG | EF | | 0000V | 28 | 00066 | | .DISPL | 58 | | |
| | | | | | 0000V | D0 | 00072 | | .DISPL | 108 | | |
| | | | | | 0000V | 11 | 0007D | | .DISPL | 118 | | |
| | | | | | 0000V | 00 | 0007F | 58: | .DISPL | 98 | | |
| | | | | | 0000V | 28 | 00087 | | BRW | 128 | | |
| | | | | | 0000V | 11 | 00093 | | MOV C3 | #32,C.AAP,Y LABEL | : 3552 | |
| | | | | | 0000V | D0 | 000A1 | 88: | MOVL | DATA+20,DATA+172 | : 3553 | |
| | | | | | 0000V | 11 | 000AC | | BRB | 138 | | |
| | | | | | 0000V | 28 | 000AE | 98: | BBC | #0,VARIABLE RECORDS,78 | : 3561 | |
| | | | | | 0000V | D0 | 000BA | | MOV C3 | #32,C.AAQ,Y LABEL | : 3563 | |
| | | | | | 0000V | 11 | 000C5 | | BRB | 88 | | |
| | | | | | 0000V | 28 | 000C7 | 108: | MOV C3 | #32,C.AAR,Y LABEL | : 3567 | |
| | | | | | 0000V | D0 | 000D3 | | MOVL | DATA+20,DATA+232 | : 3569 | |
| | | | | | 0000V | 11 | 000DE | | BRB | 138 | | |
| | | | | | 0000V | 28 | 000E0 | 118: | MOV C3 | #32,C.AAS,Y LABEL | : 3577 | |
| | | | | | 0000V | D0 | 000EC | | MOVL | DATA+20,DATA+216 | : 3578 | |
| | | | | | 0000V | 11 | 000F7 | | BRB | 138 | | |
| | | | | | 0000V | D4 | 000F9 | 128: | MOV C3 | #32,C.AAT,Y LABEL | : 3586 | |
| | | | | | 0000V | D0 | 000FB | 148: | MOVL | DATA+20,DATA+192 | : 3587 | |
| | | | | | 0000V | C5 | 000FE | | BRB | 138 | | |
| | | | | | 0000V | D4 | 00102 | | MOV C3 | #32,C.AAU,Y LABEL | : 3595 | |
| | | | | | 0000V | D0 | 00104 | 158: | MOVL | DATA+20,DATA+136 | : 3596 | |
| | | | | | 0000V | C1 | 00107 | | BRB | 138 | | |
| | | | | | 0000V | C1 | 0010F | | CLRL | R12 | : 3606 | |
| | | | | | 0000V | 00 | 00113 | | MOVL | R12,I | | |
| | | | | | 0000V | 50 | 00118 | | MULL3 | #32,I,R6 | : 3610 | |
| | | | | | 0000V | 1F | 00120 | | CLRL | R7 | | |
| | | | | | 0000V | 00 | 00124 | | MOV | R7,J | | |
| | | | | | 0000V | 50 | 00129 | | ADDL3 | #1,J,DATA+148 | : 3617 | |
| | | | | | 0000V | 20 | 0012C | | ADDL3 | R6,J,R8 | : 3618 | |
| | | | | | 0000V | 56 | 00130 | | CALLS | #0,PROLOGUE3 DEPTH | | |
| | | | | | 0000V | D0 | 00132 | 168: | MOVL | R0,XY PLOT[R8] | | |
| | | | | | 0000V | C1 | 00135 | | AOBLEQ | #31,R7,158 | | |
| | | | | | 0000V | 9A | 00139 | | CALLS | #0,NATURAL DEPTH | : 3625 | |
| | | | | | 0000V | F3 | 00146 | | MOVL | R0,TEMP INTEGER | | |
| | | | | | 0000V | CF | 0014A | | MULL3 | #32,I,R0 | : 3627 | |
| | | | | | 0000V | 0C | 00152 | | CLRL | R6 | | |
| | | | | | 0000V | 00 | 00154 | | MOV | R6,TEMP_INT2 | | |
| | | | | | 0000V | 00 | 00156 | | ADDL3 | R0,TEMP_INT2,R7 | : 3629 | |
| | | | | | 0000V | 00 | 00158 | | MOVZBL | COLOR_ROW[TEMP_INT2],COLOR_PLOT[R7] | | |
| | | | | | 0000V | 00 | 0015A | | AOBLEQ | #31,R8,168 | | |
| | | | | | 0000V | 00 | 0015C | | CASEL | DATA+280,#0,#4 | : 3631 | |
| | | | | | 0000V | 11 | 0015E | 178: | .DISPL | 178 | | |
| | | | | | 0000V | C0 | 00169 | | .DISPL | 188 | | |
| | | | | | 0000V | 11 | 0016B | 188: | .DISPL | 208 | | |
| | | | | | 0000V | C0 | 00176 | | .DISPL | 218 | | |
| | | | | | 0000V | 11 | 00178 | 198: | .DISPL | 218 | | |
| | | | | | 0000V | C0 | 00183 | | .DISPL | 198 | | |
| | | | | | 0000V | C0 | 00185 | 208: | BRB | 228 | | |
| | | | | | 0000V | 11 | 00185 | | ADDL2 | DATA+24,DATA+172 | : 3633 | |
| | | | | | 0000V | 11 | 00185 | | BRB | 238 | | |
| | | | | | 0000V | C0 | 00185 | 188: | ADDL2 | DATA+24,DATA+232 | : 3636 | |
| | | | | | 0000V | 11 | 00185 | | BRB | 238 | | |
| | | | | | 0000V | C0 | 00185 | 198: | ADDL2 | DATA+24,DATA+216 | : 3639 | |
| | | | | | 0000V | 11 | 00185 | 208: | BRB | 238 | | |
| | | | | | 0000V | C0 | 00185 | | ADDL2 | DATA+24,DATA+192 | : 3642 | |

FF56

5C

01

| | | | | | | |
|-----|-------|-------|-------|---------------------|--|--------|
| 00V | 11 | 00190 | BRB | 23\$ | | |
| EF | C0 | 00192 | ADDL2 | IDATA+24, IDATA+136 | | ; 3645 |
| 00V | 11 | 0019D | BRB | 23\$ | | |
| | | 0019F | | | | |
| 0C | F1 | 0019F | ACBL | #12, #1, R12, 14\$ | | |
| 04 | 001A5 | RET | | | | ; 3656 |

; Routine Size: 422 bytes, Routine Base: \$CODE + 021EC

| | | | | | | |
|--------------|-----------|-----------|-------|--------------|--------------|--------------------|
| | | 00000 | 00000 | SETUP_GRAPH: | | ; 3702 |
| | | 00000 | 00000 | .WORD | ^M<> | |
| | 00000014G | EF | D4 | 00002 | CLRL | IDATA+20 |
| | 00000010G | EF | D4 | 00008 | CLRL | IDATA+16 |
| | 00000018G | EF | D4 | 0000E | CLRL | IDATA+24 |
| 00V00000000G | EF | 00 | E0 | 00014 | BBS | #0, AUTO_TUNE, 2\$ |
| | 00000000G | EF | 9F | 0001C | PUSHAB | PASS\$FV OUTPUT |
| | 00000000G | EF | 01 | FB | 00022 | CALLS |
| | 02 | 00000118G | EF | D1 | 00029 | 2\$: CMPL |
| | | 00V | 12 | 00030 | BNEQ | 6\$ |
| | 000000031 | 8F | DF | 00032 | PUSHAL | #49 |
| 00000000G | EF | 01 | FB | 00038 | CALLS | #1, QUERY |
| | 000000032 | 8F | DF | 0003F | PUSHAL | #50 |
| 00000000G | EF | 01 | FB | 00045 | CALLS | #1, QUERY |
| | 3B9AC9FF | 8F | DF | 0004C | PUSHAL | #999999999 |
| | 000000000 | 8F | DF | 00052 | PUSHAL | #0 |
| 00000000G | EF | 02 | FB | 00058 | CALLS | #2, AUTO_SCALE |
| | | 00V | 11 | 0005F | BRB | 8\$ |
| | 000000030 | 8F | DF | 00061 | 6\$: PUSHAL | #48 |
| 00000000G | EF | 01 | FB | 00067 | CALLS | #1, QUERY |
| | 000000038 | 8F | DF | 0006E | 8\$: PUSHAL | #56 |
| 00000000G | EF | 01 | FB | 00074 | CALLS | #1, QUERY |
| | 000000017 | 8F | DF | 0007B | PUSHAL | #23 |
| 00000000G | EF | 01 | FB | 00081 | CALLS | #1, QUERY |
| | 03 | 00000118G | EF | D1 | 00088 | CMPL |
| | | 00V | 12 | 0008F | BNEQ | 14\$ |
| | 000000023 | 8F | DF | 00091 | PUSHAL | #35 |
| 00000000G | EF | 01 | FB | 00097 | CALLS | #1, QUERY |
| | 000000024 | 8F | DF | 0009E | PUSHAL | #36 |
| 00000000G | EF | 01 | FB | 000A4 | CALLS | #1, QUERY |
| | 3B9AC9FF | 8F | DF | 000AB | PUSHAL | #999999999 |
| | 000000000 | 8F | DF | 000B1 | PUSHAL | #0 |
| 00000000G | EF | 02 | FB | 000B7 | CALLS | #2, AUTO_SCALE |
| | | 00V | 11 | 000BE | BRB | 16\$ |
| | 000000022 | 8F | DF | 000C0 | 14\$: PUSHAL | #34 |
| 00000000G | EF | 01 | FB | 000C6 | CALLS | #1, QUERY |
| | 000000016 | 8F | DF | 000CD | 16\$: PUSHAL | #22 |
| 00000000G | EF | 01 | FB | 000D3 | CALLS | #1, QUERY |
| | 00000001D | 8F | DF | 000DA | PUSHAL | #29 |
| 00000000G | EF | 01 | FB | 000E0 | CALLS | #1, QUERY |
| | 00000118G | EF | D5 | 000E7 | TSTL | IDATA+280 |
| | | 00V | 12 | 000ED | BNEQ | 22\$ |
| | 00000002C | 8F | DF | 000EF | PUSHAL | #44 |
| 00000000G | EF | 01 | FB | 000F5 | CALLS | #1, QUERY |
| | 00000002D | 8F | DF | 000FC | PUSHAL | #45 |
| 00000000G | EF | 01 | FB | 00102 | CALLS | #1, QUERY |
| | 000000064 | 8F | DF | 00109 | PUSHAL | #100 |
| | 00000001F | 8F | DF | 0010F | PUSHAL | #31 |

| Generated Code | | | | | | | |
|---|----|-----------|-----|-------|-----------|---------------|--------|
| 00000000G | EF | 02 | FB | 00115 | CALLS | #2,AUTO_SCALE | |
| | | 00V | 11 | 0011C | BRB | 24\$ | |
| 00000000G | EF | 0000002B | 8F | DF | 0011E | 22\$: PUSHAL | : 3766 |
| 00000000G | EF | 00000040 | 01 | FB | 00124 | CALLS | : 3768 |
| 00000000G | EF | 00000118G | 8F | DF | 0012B | 24\$: PUSHAL | : 3770 |
| | 01 | | 01 | FB | 00131 | CALLS | |
| | | | EF | D1 | 00138 | CMPL | |
| | | | 00V | 12 | 0013F | BNEQ | |
| 00000000G | EF | 00000044 | 8F | DF | 00141 | PUSHAL | : 3774 |
| 00000000G | EF | 00000045 | 01 | FB | 00147 | CALLS | : 3775 |
| 00000000G | EF | 00000000G | 8F | DF | 0014E | PUSHAL | : 3776 |
| | | 00000001 | 01 | FB | 00154 | CALLS | |
| | | | EF | 9F | 0015B | PUSHAB | |
| | | | 8F | DF | 00161 | PUSHAL | |
| 00000000G | EF | 00000010G | 02 | FB | 00167 | CALLS | : 3777 |
| 000000E4G | EF | | EF | D0 | 0016E | MOVL | |
| | | | 00V | 11 | 00179 | BRB | |
| 00000000G | EF | 00000037 | 00 | FB | 0017B | 29\$: CALLS | : 3783 |
| | | | 8F | DF | 00182 | 30\$: PUSHAL | : 3785 |
| 00000000G | EF | 0000001A | 01 | FB | 00188 | CALLS | : 3786 |
| 00000000G | EF | 00000000G | 8F | DF | 0018F | PUSHAL | : 3787 |
| | | 00000118G | 01 | FB | 00195 | CALLS | : 3789 |
| | 04 | | EF | D4 | 0019C | CLRL | |
| | | | EF | D1 | 001A2 | CMPL | |
| | | | 00V | 12 | 001A9 | BNEQ | |
| 00000000G | EF | 00000034 | 8F | DF | 001AB | PUSHAL | : 3793 |
| | | | 01 | FB | 001B1 | CALLS | |
| 00000000G | EF | 00000035 | 8F | DF | 001B8 | PUSHAL | : 3794 |
| | | | 01 | FB | 001BE | CALLS | |
| | | | EF | 9F | 001C5 | PUSHAB | |
| | | | 8F | DF | 001CB | PUSHAL | |
| 00000000G | EF | 0000003E | 02 | FB | 001D1 | CALLS | : 3795 |
| | | | 00V | 11 | 001D8 | BRB | |
| 00000000G | EF | | 00 | FB | 001DA | 36\$: CALLS | : 3801 |
| 00000000G | EF | | 00 | FB | 001E1 | 37\$: CALLS | : 3803 |
| 00000000G | EF | | 00 | FB | 001E8 | CALLS | : 3804 |
| | | | 8F | DF | 001EF | PUSHAL | : 3805 |
| 00000000G | EF | | 01 | FB | 001F5 | CALLS | |
| 00000000G | EF | | 00 | FB | 001FC | CALLS | : 3806 |
| 00000000G | EF | | 00 | FB | 00203 | CALLS | : 3807 |
| 00000000G | EF | | 00 | FB | 0020A | CALLS | : 3808 |
| 00V00000000G | EF | | 00 | EO | 00211 | BBS | : 3810 |
| | | | EF | 9F | 00219 | PUSHAB | : 3812 |
| 00000000G | EF | 00000000G | 01 | FB | 0021F | CALLS | |
| | 05 | 00000118G | EF | D1 | 00226 | 40\$: CMPL | : 3819 |
| | | | 00V | 13 | 0022D | BEQL | |
| 21EC | CF | | 00 | FB | 0022F | CALLS | : 3824 |
| | | | 04 | 00234 | 42\$: RET | | : 3826 |
| ; Routine Size: 565 bytes, Routine Base: \$CODE + 02392 | | | | | | | |
| 00000000G EF 00000046 8F DF 00002 00000 PLOT_AND_DESIGN: : 3876 | | | | | | | |
| 00000000G EF 00000000 01 FB 00008 -WORD ^M<> : 3883 | | | | | | | |
| 00000000G EF 00000000 00 FB 0000F CALLS #70 : 3889 | | | | | | | |
| 00000000G EF 00000000 01 90 00014 CALLS #1,QUERY : 3890 | | | | | | | |
| 00000000G EF 00000000 00 FB 0000F CALLS #0,SETUP_GRAPH : 3889 | | | | | | | |
| 00000000G EF 00000000 01 90 00014 MOV B #1,VISIBL_QUESTION : 3890 | | | | | | | |

Generated Code

E 13
16-Sep-1984 01:10:30
5-Sep-1984 13:36:36VAX-11 Pascal V2.4-277
DISK\$VMSMASTER:[EDF.SRC]EDFDESIGN.PAS;1 (38) Page 122

| | | | | | | | | | |
|-----------|----|-----------|-------|----|-------|--------|--------------------------|---|------|
| 00000000G | EF | 00000000G | EF | 90 | 0001B | MOVB | AUTO TUNE, TAKE_DEFAULTS | : | 3891 |
| | | 00000000G | EF | 9F | 00026 | PUSHAB | LINES_PER_PAGE | : | 3896 |
| | | 00000000G | EF | 9F | 0002C | PUSHAB | PROMPT_LINE | : | |
| 00000000G | EF | | 02 | FB | 00032 | CALLS | #2, LIB\$SET_SCROLL | : | |
| 00000000G | EF | | 01 | 90 | 00039 | MOVB | #1, SCROLLING_SET | : | 3897 |
| 00000000G | EF | | 01 | 90 | 00040 | MOVB | #1, WAIT_HELP | : | 3898 |
| 00000000G | EF | | 01 | 90 | 00047 | MOVB | #1, FIRST_PLOT | : | 3903 |
| 0A1A | CF | | 00 | FB | 0004E | CALLS | #0, PLOT_GRAPH | : | 3908 |
| | | 00000000G | EF | 94 | 00053 | CLRB | LINKED | : | 3914 |
| | | | 0000V | 31 | 00059 | BRW | 37\$ | : | 3916 |
| | | 0000002A | BF | DF | 0005C | PUSHAL | #42 | : | 3923 |
| 3F | EF | 00000000G | 01 | FB | 00062 | CALLS | #1, QUERY | : | |
| | | 000000A8G | EF | CF | 00069 | CASEL | IDATA+168, #1, #63 | : | 3925 |
| | | | 0000V | | 00071 | .DISPL | 28\$ | : | |
| | | | 0080 | | 00073 | .DISPL | 128 | : | |
| | | | 0000V | | 00075 | .DISPL | 29\$ | : | |
| | | | 0080 | | 00077 | .DISPL | 128 | : | |
| | | | 0080 | | 00079 | .DISPL | 128 | : | |
| | | | 0080 | | 0007B | .DISPL | 128 | : | |
| | | | 0080 | | 0007D | .DISPL | 128 | : | |
| | | | 0080 | | 0007F | .DISPL | 128 | : | |
| | | | 0080 | | 00081 | .DISPL | 128 | : | |
| | | | 0080 | | 00083 | .DISPL | 128 | : | |
| | | | 0080 | | 00085 | .DISPL | 128 | : | |
| | | | 0080 | | 00087 | .DISPL | 128 | : | |
| | | | 0080 | | 00089 | .DISPL | 128 | : | |
| | | | 0080 | | 0008B | .DISPL | 128 | : | |
| | | | 0080 | | 0008D | .DISPL | 128 | : | |
| | | | 0000V | | 0008F | .DISPL | 22\$ | : | |
| | | | 0000V | | 00091 | .DISPL | 21\$ | : | |
| | | | 0000V | | 00093 | .DISPL | 23\$ | : | |
| | | | 0080 | | 00095 | .DISPL | 128 | : | |
| | | | 0080 | | 00097 | .DISPL | 128 | : | |
| | | | 0080 | | 00099 | .DISPL | 128 | : | |
| | | | 0080 | | 0009B | .DISPL | 128 | : | |
| | | | 0080 | | 0009D | .DISPL | 128 | : | |
| | | | 0080 | | 0009F | .DISPL | 128 | : | |
| | | | 0080 | | 000A1 | .DISPL | 128 | : | |
| | | | 0080 | | 000A3 | .DISPL | 128 | : | |
| | | | 0080 | | 000A5 | .DISPL | 128 | : | |
| | | | 0080 | | 000A7 | .DISPL | 128 | : | |
| | | | 0080 | | 000A9 | .DISPL | 128 | : | |
| | | | 0000V | | 000AB | .DISPL | 20\$ | : | |
| | | | 0080 | | 000AD | .DISPL | 128 | : | |
| | | | 0080 | | 000AF | .DISPL | 128 | : | |
| | | | 0080 | | 000B1 | .DISPL | 128 | : | |
| | | | 0000V | | 000B3 | .DISPL | 18\$ | : | |
| | | | 0080 | | 000B5 | .DISPL | 128 | : | |
| | | | 0080 | | 000B7 | .DISPL | 128 | : | |
| | | | 0080 | | 000B9 | .DISPL | 128 | : | |
| | | | 0000V | | 000BB | .DISPL | 11\$ | : | |
| | | | 0080 | | 000BD | .DISPL | 128 | : | |
| | | | 0080 | | 000BF | .DISPL | 128 | : | |
| | | | 0080 | | 000C1 | .DISPL | 128 | : | |
| | | | 0080 | | 000C3 | .DISPL | 128 | : | |
| | | | 0000V | | 000C5 | .DISPL | 9\$ | : | |
| | | | 0080 | | 000C7 | .DISPL | 128 | : | |

| | | | | | | | | |
|-----------|----|-------|----|-------|------|--------|--------------------------|--------|
| | | 0080 | | 000C9 | | .DISPL | 128 | |
| | | 0080 | | 000CB | | .DISPL | 128 | |
| | | 0080 | | 000CD | | .DISPL | 128 | |
| | | 0000V | | 000CF | | .DISPL | 138 | |
| | | 0080 | | 000D1 | | .DISPL | 128 | |
| | | 0080 | | 000D3 | | .DISPL | 128 | |
| | | 0000V | | 000D5 | | .DISPL | 158 | |
| | | 0080 | | 000D7 | | .DISPL | 128 | |
| | | 0080 | | 000D9 | | .DISPL | 128 | |
| | | 0000V | | 000DB | | .DISPL | 88 | |
| | | 0000V | | 000DD | | .DISPL | 268 | |
| | | 0000V | | 000DF | | .DISPL | 168 | |
| | | 0080 | | 000E1 | | .DISPL | 128 | |
| | | 0000V | | 000E3 | | .DISPL | 78 | |
| | | 0080 | | 000E5 | | .DISPL | 128 | |
| | | 0080 | | 000E7 | | .DISPL | 128 | |
| | | 0080 | | 000E9 | | .DISPL | 128 | |
| | | 0000V | | 000EB | | .DISPL | 248 | |
| | | 0080 | | 000ED | | .DISPL | 128 | |
| | | 0000V | | 000EF | | .DISPL | 58 | |
| | | 0000V | 31 | 000F1 | | BRW | 308 | |
| | | 8F | DF | 000F4 | 58: | PUSHAL | #64 | : 3927 |
| 00000000G | EF | 01 | FB | 000FA | | CALLS | #1, QUERY | |
| | | 0000V | 31 | 00101 | | BRW | 318 | |
| 00000000G | EF | 00 | FB | 00104 | 78: | CALLS | #0, ASK_MEAN_RECORD_SIZE | : 3929 |
| | | 0000V | 31 | 0010B | | BRW | 318 | |
| 00000000G | EF | 00 | FB | 0010E | 88: | CALLS | #0, ASK_KEY_SIZE | : 3931 |
| | | 0000V | 31 | 00115 | | BRW | 318 | |
| | | 8F | DF | 00118 | 98: | PUSHAL | #43 | : 3933 |
| 00000000G | EF | 01 | FB | 0011E | | CALLS | #1, QUERY | |
| | | 0000V | 31 | 00125 | | BRW | 318 | |
| | | 8F | DF | 00128 | 118: | PUSHAL | #38 | : 3935 |
| 00000000G | EF | 01 | FB | 0012E | | CALLS | #1, QUERY | |
| | | 0000V | 31 | 00135 | | BRW | 318 | |
| | | 8F | DF | 00138 | 138: | PUSHAL | #48 | : 3937 |
| 00000000G | EF | 01 | FB | 0013E | | CALLS | #1, QUERY | |
| | | 00V | 11 | 00145 | | BRB | 318 | |
| 00000000G | EF | 00 | FB | 00147 | 158: | CALLS | #0, ASK_KEY_POSITION | : 3939 |
| | | 00V | 11 | 0014E | | BRB | 318 | |
| | | 8F | DF | 00150 | 168: | PUSHAL | #56 | : 3941 |
| 00000000G | EF | 01 | FB | 00156 | | CALLS | #1, QUERY | |
| | | 00V | 11 | 0015D | | BRB | 318 | |
| | | 8F | DF | 0015F | 188: | PUSHAL | #34 | : 3943 |
| 00000000G | EF | 01 | FB | 00165 | | CALLS | #1, QUERY | |
| | | 00V | 11 | 0016C | | BRB | 318 | |
| 00000000G | EF | 00 | FB | 0016E | 208: | CALLS | #0, ASK_KEY_DUPS | : 3945 |
| | | 00V | 11 | 00175 | | BRB | 318 | |
| 00000000G | EF | 00 | FB | 00177 | 218: | CALLS | #0, ASK_REC_COMP | : 3947 |
| | | 00V | 11 | 0017E | | BRB | 318 | |
| 00000000G | EF | 00 | FB | 00180 | 228: | CALLS | #0, ASK_KEY_COMP | : 3949 |
| | | 00V | 11 | 00187 | | BRB | 318 | |
| 00000000G | EF | 00 | FB | 00189 | 238: | CALLS | #0, ASK_IDX_COMP | : 3951 |
| | | 00V | 11 | 00190 | | BRB | 318 | |
| | | 8F | DF | 00192 | 248: | PUSHAL | #62 | : 3953 |
| 00000000G | EF | 01 | FB | 00198 | | CALLS | #1, QUERY | |
| | | 00V | 11 | 0019F | | BRB | 318 | |
| | | 8F | DF | 001A1 | 268: | PUSHAL | #55 | : 3955 |

```
Generated Code
00000000G EF 01 FB 001A7 CALLS #1, QUERY
1CC5 CF 00V 11 001AE BRB 31$
00000000G EF 00 FB 001B0 28$: CALLS #0, LINK_RESULTS ; 3957
0A1A CF 00V 11 001B5 BRB 31$
00000000G EF 01 90 001B7 29$: MOVB #1, FIRST_PLOT ; 3966
0A1A CF 00 FB 001BE CALLS #0, PLOT_GRAPH ; 3967
00V 11 001C3 BRB 31$
03 000000A8G EF D1 001C5 30$: CMPL IDATA+168, #3 ; 3981
00V 13 001CC 31$: BEQL 37$
00V00000000G EF 00 E0 001CE BBS #0, LINKED, 37$
05 00000118G EF D1 001D6 CMPL IDATA+280, #5 ; 3985
00V 13 001DD BEQL 35$
21EC CF 00 FB 001DF CALLS #0, CALC_ARRAY ; 3987
0A1A CF 00 FB 001E4 35$: CALLS #0, PLOT_GRAPH ; 3989
03 00000000G EF 00 E0 001E9 37$: BBS #0, LINKED, .+3
FE68 31 001F1 BRW 3$
00000000G EF 00 FB 001F4 CALLS #0, EDF$RESET_SCROLL ; 3995
04 001FB RET ; 3997
```

; Routine Size: 508 bytes, Routine Base: \$CODE + 025C7

```
00000000G EF 0000003D 8F 0000 SEQ_REL_WORK: ; 4043
00000000G EF 00000040 8F DF 00002 .WORD ; 4050
00000000G EF 00000018 01 FB 00008 PUSHAL #M<> ; 4051
00000000G EF 00000018 8F DF 0000F PUSHAL #61 ; 4052
00000000G EF 00000018 01 FB 00015 CALLS #1, QUERY ; 4053
00000000G EF 00000018 8F DF 0001C PUSHAL #64 ; 4054
00000000G EF 00000018 01 FB 00022 CALLS #1, QUERY ; 4055
00000000G EF 00000018 00 FB 00029 CALLS #0, ASK_MEAN_RECORD_SIZE ; 4056
00000000G EF 00000018 00 FB 00030 CALLS #0, INIT_DEF ; 4057
0D02 CF 00 FB 00037 CALLS #0, NON_KEY_DEF ; 4058
04 0003C RET ; 4059
```

; Routine Size: 61 bytes, Routine Base: \$CODE + 027C3

```
00000000G EF 00000020 01 FB 00010 INDEXED_DESIGN: ; 4112
00V00000000G EF 00000020 00 E0 00017 .WORD ; 4124
00V 52 04 BC 90 00002 MOVB @4(R12), REDESIGN_FLAG ; 4129
00V 5C 08 BC 90 00006 MOVB @8(R12), ADD_KEY_FLAG ; 4133
00000000G EF 00000020 8F DF 0000A PUSHAL #32 ; 4140
00V 52 04 BC 90 00010 CALLS #1, QUERY ; 4142
00V 5C 08 BC 90 00017 BBS #0, OPTIMIZING, 10$ ; 4144
00000000G EF 00000021 8F DF 00025 BLBC REDESIGN_FLAG, 7$ ; 4145
00V 5C 08 BC 90 00022 BLBS ADD_KEY_FLAG, 6$ ; 4153
00000000G EF 00000021 8F DF 00025 PUSHAL #33 ; 4154
00V 5C 08 BC 90 00022 CALLS #1, QUERY ; 4155
00000000G EF 00000021 01 FB 0002B 6$: MOVL IDATA+132, BEGINING_KEY ; 4156
00V 5C 08 BC 90 00022 7$: MOVL BEGINING_KEY, ENDING_KEY ; 4157
00000000G EF 00000021 8F DF 0003E 7$: BRB 11$ ; 4158
00V 5C 08 BC 90 0003E 7$: PUSHAL #60 ; 4159
00000000G EF 00000021 01 FB 00044 CALLS #1, QUERY ; 4160
00V 5C 08 BC 90 00044 CLRL BEGINING_KEY ; 4161
00000000G EF 00000021 01 C3 0004D 10$: SUBL3 #1, IDATA+240, ENDING_KEY ; 4162
00V 5C 08 BC 90 0004D 10$: BRB 11$ ; 4163
00000000G EF 00000021 8F 9F 00057 10$: PUSHAB #1 ; 4164
```

| | | | | | | | | | |
|-----------|----|--------------|-----------|-----|----|-------|----------------|----------------------------|--------|
| 000000F0G | EF | 00000000G | EF | 01 | FB | 0005A | CALLS | #1,SCAN_DEFINITION | |
| | | | | 01 | C1 | 00061 | ADDL3 | #1,HIGH_KEY,IDATA+240 | : 4166 |
| | | | | 53 | D4 | 0006D | CLRL | BEGINING_KEY | : 4167 |
| | | 54 | 00000000G | EF | D0 | 0006F | MOVL | HIGH_KEY,ENDING_KEY | : 4168 |
| | | 54 | | 53 | D1 | 00076 | 11\$: CMPL | R3,R4 | : 4175 |
| | | | | 00V | 14 | 00079 | BGTR | 16\$ | |
| | | 52 | | 5C | 8A | 0007B | BICB2 | ADD_KEY_FLAG,REDESIGN_FLAG | |
| | | | | 00V | 11 | 0007E | BRB | 13\$ | |
| | | | | 53 | D6 | 00080 | 12\$: INCL | R3 | |
| | | 00000084G | 5C | 53 | D0 | 00082 | 13\$: MOVL | R3,ACTIVE_KEY_INDEX | |
| | | | EF | 5C | D0 | 00085 | MOVL | ACTIVE_KEY_INDEX,IDATA+132 | : 4179 |
| | | | 00V | 52 | E9 | 0008C | BLBC | R2,15\$ | : 4181 |
| | | 0BB4 | CF | 00 | FB | 0008F | CALLS | #0,WARN_OF_ERASE | : 4187 |
| | | 25C7 | CF | 00 | FB | 00094 | 15\$: CALLS | #0,PLOT_AND_DESIGN | : 4189 |
| | | | 54 | 53 | D1 | 00099 | CMPL | R3,R4 | |
| | | | | E2 | 19 | 0009C | BLSS | 12\$ | |
| | | 00V00000000G | EF | 00 | E1 | 0009E | 16\$: BBC | #0,AUTO_TUNE,18\$ | : 4197 |
| | | 000002F3G | EF | 02 | D0 | 000A6 | MOVL | #2,QTAB*755 | : 4199 |
| | | | | | 04 | 000AD | 18\$: RET | | : 4201 |

; Routine Size: 174 bytes, Routine Base: \$CODE + 02800

028AE .END

EDFDESIGN
V04-000

Pascal Compilation Statistics

I 13
16-Sep-1984 01:10:30
5-Sep-1984 13:36:36

VAX-11 Pascal V2.4-277
DISK\$VMSMASTER:[EDF.SRC]EDFDESIGN.PAS;1 (38)
Page 126

COMMAND QUALIFIERS

PASCAL/MACHINE/NODEBUG/NOCHECK/LIS=LIS\$:EDFDESIGN/OBJ=OBJ\$:EDFDESIGN MSRC\$:EDFDESIGN

/CHECK=(NOBOUNDS,NOCASE_SELECTORS,NOOVERFLOW,NOPOINTERS,NOSUBRANGE)

/DEBUG=(NOSYMBOLS,NOTRACEBACK)

/ENVIRONMENT= \$255\$DUA28:[EDF.OBJ]EDFDESIGN.PEN;1

/LIST= \$255\$DUA28:[EDF.LIS]EDFDESIGN.LIS;1

/OBJECT= \$255\$DUA28:[EDF.OBJ]EDFDESIGN.OBJ;1

/NOCROSS_REFERENCE /ERROR_LIMIT=30 /NOG_FLOATING /MACHINE_CODE /NOOLD_VERSION /OPTIMIZE /NOSTANDARD /WARNINGS

COMPILER INTERNAL TIMING

| Phase | Faults | CPU Time | Elapsed Time |
|-------------------|--------|----------|--------------|
| Initialization | 95 | 00:00.5 | 00:03.3 |
| Source Analysis | 2039 | 00:25.8 | 04:44.4 |
| Source Listing | 55 | 00:05.2 | 00:11.5 |
| Tree Construction | 513 | 00:03.0 | 00:08.8 |
| Flow Analysis | 102 | 00:01.5 | 00:03.4 |
| Profit Analysis | 233 | 00:02.2 | 00:05.0 |
| Context Analysis | 503 | 00:20.1 | 00:42.1 |
| Name Packing | 21 | 00:00.6 | 00:01.2 |
| Code Selection | 201 | 00:03.1 | 00:06.3 |
| Final | 288 | 00:10.3 | 00:24.8 |
| TOTAL | 4058 | 01:12.4 | 06:31.0 |

COMPILATION STATISTICS

CPU Time: 01:12.4 (3483 Lines/Minute)
Elapsed Time: 06:31.0
Page Faults: 4058
Compilation Complete

0126

AH-BT13A-SE
VAX/VMS V4.

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY